UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À
L'UNIVERSITÉ DU QUÉBEC À TROIS-RIVIÈRES

COMME EXIGENCE PARTIELLE
DE LA MAÎTRISE EN MATHÉMATIQUES ET INFORMATIQUE APPLIQUÉE

PAR
ANDRÉ BOUMSO

MÉTHODE EXPLORATOIRE DE DISTRIBUTION DES CLÉS DE CRYPTAGE POUR LES COMMUNICATIONS DE GROUPE DANS UN RÉSEAU MOBILE AD HOC

AVRIL 2006

ABSTRACT

Multicast and mobile communications are emergent technologies that might enable interactive real time applications such as video on demand and videoconference to name a few. In infrastructureless zones, ad hoc networks seem to be a natural extension of networks with fixed infrastructures. However, securing group's communications in these networks is very challenging, since they suffer cruelly from a lack of resources and significant topology changes.

In this work we address the problem of Multicast secure data over a multihop wireless ad hoc network. Many key distribution protocols that have been proposed are not really convenient for mobile ad hoc networks. We propose a key agreement protocol that aims to solve problems that are specific to ad hoc networks such as mobility, unreliable links, and multihop communication cost. The main idea is to focus on group dynamics and complete node mobility in ad hoc environment to develop an adaptive protocol that is suitable for the network and group changes.

Doing so, we extend and adapt the proposed Tree based Group Diffie-Hellman (TGDH) protocol to pure mobile ad hoc network. The result of this work is the development of the Group Activity based Key Agreement Protocol (GAKAP).

RÉSUMÉ

Le multicast et les communications mobiles sont des technologies émergentes permettant l'utilisation des applications en temps réel tel que la vidéoconférence, la vidéo sur demande. Dans les zones ne disposant pas d'infrastructures de communication, les réseaux mobiles ad hoc apparaissent comme le complément idéal pour étendre les réseaux à infrastructures fixes. Cependant, la sécurité des communications de groupe dans les réseaux ad hoc est une tâche ardue et complexe à cause du manque de ressources et du changement fréquent de la topologie du réseau.

Dans notre travail, nous tentons d'apporter une solution à la sécurité des communications de groupe dans les réseaux ad hoc. En effet, plusieurs autres solutions de distribution des clés de chiffrement lors des communications de groupe ont déjà été proposées mais elles se sont avérées limitées car elles ne tiennent pas compte des effets conjugués des conséquences de la mobilité des nœuds et de la dynamique du groupe dans les réseaux ad hoc.

Nous proposons le protocole GAKAP (Group Activity based Key Agreement Protocol), un protocole d'accord de clés de chiffrement qui, en tenant compte à la fois de la mobilité et des opérations de groupe dans un réseau ad hoc très dynamique, tente de réduire les coûts reliés à la consommation des ressources nécessaires à la distribution de clés pour les communications multicast.

Pour le développement du protocole GAKAP, nous avons étendu et adapté le protocole Tree based Group Diffie-Hellman (TGDH) aux communications de groupe dans un réseau mobile ad hoc pur.

REMERCIEMENTS

Je remercie sincèrement tous les évaluateurs de ce mémoire, les professeurs François Meunier et Mhamed Messfoui, qui ont bien voulu donner de leur temps pour le lire et suggérer les correctifs nécessaires à son approbation comme document scientifique.

Je remercie mes professeurs Boucif Amar Bensaber et Ismaïl Biskri qui ont accepté de travailler avec moi sur ce projet et m'ont permis de rédiger et de participer à la présentation de ce travail lors de différents congrès.

Je tiens à remercier le professeur Mhamed Messfoui de sa disponibilité à répondre à différentes questions statistiques que je lui ai posées à de nombreuses occasions.

Je remercie ma mère Ngo Boumso Augustine d'avoir consenti, malgré ses moyens très limités, tous les sacrifices nous ayant permis mes frères et moi d'aspirer à une meilleure instruction.

Je remercie mon épouse qui a su m'épauler, malgré tous les sacrifices que la rédaction d'un mémoire comporte, à parvenir au terme de ce projet.

J'adresse ma reconnaissance à ma sœur Monique Mpeck, à son époux Philippe et à tous ceux qui à un moment ou à un autre ont su me donner une chance.

Je remercie Guylaine Cossette qui a accepté de corriger ce mémoire et Pauline Bordeleau d'avoir alléger mes heures « d'accouchement ».

TABLE DES MATIÈRES

LISTE DES TABLEAUX

LISTE DES FIGURES

Chapitre I

LES COMMUNICATIONS DE GROUPE : LE MULTICAST

1.1 Introduction

Les communications ont toujours été au centre des rapports entre personnes et entre communautés. Depuis les colonnes de fumées des indiens, l'écho des tambours des africains, le courriel en passant par le courrier traditionnel et le téléphone, les moyens de communication ont pris diverses formes et se sont raffinées dans le temps. Les télécommunications connaissent une croissance phénoménale, les réseaux informatiques aussi. L'avènement de l'Internet est une véritable révolution dans le monde des média car il a ouvert une grande vitrine sur le monde et servi comme base de lancement à de nouvelles applications de masse comme le multimédia. Mais ces applications sont généralement soutenues par un mode de communication de point à point. Or ce mode de communication a pour conséquence, entre autre, une importante consommation de la bande passante. Il fallait trouver des méthodes de communication de masse plus efficaces et le multicast est une des techniques de communication qui sont en plein essor.

1.2 Les formes de communication

Dans un réseau, les ordinateurs peuvent communiquer de façon directe ou point à point, ou par diffusion. Par ailleurs, il existe deux formes de diffusion : la diffusion large (broadcast) ou la diffusion restreinte (multicast).

- Communication Unicast : c'est l'envoie d'un message d'un émetteur vers un seul récepteur. On parle aussi de communication point à point.

- Communication Broadcast : encore appelée diffusion large, le broadcast est l'envoie de messages à **tous** les récepteurs d'un même réseau ou d'un sous réseau donné sans que ces derniers ne le sollicitent [21].

- Communication Multicast : c'est une diffusion restreinte et consiste dans l'envoi de messages (données) d'un émetteur à plusieurs récepteurs ou de plusieurs émetteurs vers plusieurs récepteurs appartenant à un ou plusieurs **groupes**.

Lors de la communication multicast, un émetteur ou plusieurs émetteurs transmettent un unique message à plusieurs récepteurs. Le paquet multicast est transmis et acheminé par des routeurs multicast et unicast constituant un réseau fédérateur nommé le **MBone**. L'ensemble des récepteurs ou des destinataires constitue le **Groupe** [31]. On dit alors du multicast qu'il est une communication de groupe. Le groupe de communication peut être ouvert à tous ou restreint à un ensemble de membres autorisés.

Pour qu'une session multicast puisse avoir lieu, il faut d'abord que le groupe soit mis sur pieds et que les rôles des différents membres soient définis.

1.3 Description des entités

Lors d'une session multicast, on identifie les entités suivantes dépendamment de leur rôle :

- L'Initiateur : c'est l'entité qui débute la session multicast et qui détermine les paramètres ou besoins de ladite session :
  - les attributs de sécurité;
  - la liste des participants;
  - les caractéristiques des connexions;
  - les caractéristiques des participants.

La liste des participants est gérée par l'émetteur ou par une entité différente choisie par l'initiateur.

- Le Contrôleur ou gestionnaire : C'est une entité dont le rôle est d'ajouter ou de supprimer les participants, de contrôler l'accès sur la base des attributs de sécurité de la session en cours et de distribuer la clef de groupe (Trafic Encryption Key, TEK)

- Émetteur : Tout membre du groupe autorisé à transmettre les données.
- Récepteur : Tout membre du groupe autorisé à recevoir les informations transmises.
- Clé : Une clé est un nombre ou une structure nécessaire pour décoder un texte crypté. Le groupe utilise plusieurs types de clés, publiques ou privées, dépendamment des besoins.
- Message : C'est un unique module d'information envoyé d'un membre du groupe à un autre [20].
- Échange : Envoie réciproque de message entre deux membres d'un même groupe [20].
- Diffusion : Message envoyé par un membre aux autres membres du groupe [20].

1.4 Étapes de la création du groupe multicast

En général, la création d'un groupe multicast s'effectue en cinq étapes : l'initialisation du groupe, la sélection du contrôleur, la spécification des membres du groupe, la distribution de la liste des participants et la génération de la clef de groupe.

- L'Initialisation du groupe : elle est effectuée par l'**Initiateur** du groupe. Il définit les attributs de sécurité et crée la liste de contrôle d'accès du groupe.
- Choix du contrôleur de groupe : l'Initiateur choisit un **Contrôleur** de groupe dont le rôle est d'ajouter ou de supprimer les participants, de contrôler l'accès sur la base des attributs de sécurité de la session en cours, de distribuer la clef de groupe.

- La spécification des participants : elle permet la définition des autres participants du groupe autres que l'**Initiateur** et le **Contrôleur**. Ce sont ceux qui sont émetteur ou récepteur ou les deux.
- Distribution de la liste des participants: la liste des participants est remise à chacun des participants après vérification de la satisfaction des attributs et des contraintes de sécurité.
- Création de la clé de groupe : *le* contrôleur crée la clé et l'envoie à tous les membres inscrits à la session en cours. Elle permet de décrypter les messages transmis en multicast.

1.5 La gestion du groupe multicast : le protocole IGMP

Le protocole IGMP, **Internet Group Management Protocol**, est le protocole Internet de gestion des groupes qui permet aux hôtes de se joindre ou de se retirer du groupe multidiffusion. IGMP définit le dialogue entre les routeurs multi diffusion et les hôtes d'un sous réseau multicast relié à un routeur. Il vérifie également la présence des groupes actifs en s'assurant de la présence des participants dudit groupe à la session multicast en cours. Dans l'ensemble, on distingue trois versions d'IGMP : IGMPv1, IGMPv2 et IGMPv3 [31]. Ces versions permettant respectivement:

- de déterminer les groupes actifs : le routeur envoie des messages ECHO pour s'assurer de la présence d'au moins un membre dans le groupe,
- d'adresser certaines requêtes spécifiques à des groupes précis,
- aux hôtes de choisir la source de laquelle ils recevront leurs messages,
- de spécifier les sources dont ils ne veulent pas recevoir les messages.

Lorsqu'un membre veut se joindre au groupe, il envoie une requête d'ajout *JOIN* au routeur multicast de son sous réseau délégué à cet effet.

1.6 Les applications du multicast

Le multicast peut être utilisé dans différentes applications où participent plusieurs hôtes. Le tableau ci-dessous (tableau 1) présente différentes applications pouvant utiliser le multicast lors de communications entre systèmes informatiques.

	Temps réel	Non temps réel
Multimédia	Serveur vidéo Vidéo conférence Internet audio Événements multimédias	Réplication Livraison sur demande
Données seulement	Côtes de la bourse Nouvelles Tableau blanc Jeux en ligne Cache web	Acheminement des données Réplication de la base de données

Tableau 1 : Les applications du multicast

1.7 Problématique de la Sécurité des communications Multicast

La sécurité est une question préoccupante dans les communications de groupe. Les problèmes de sécurité déjà recensés dans les communications unicast se rencontrent également ici :

- le déni de service,
- le vol d'informations,
- l'écoute clandestine, etc.

Cependant, les facteurs liés à la spécificité du groupe rendent difficilement applicables les solutions déjà établies dans les communications point a point. Dès lors, l'établissement d'une solution de sécurité dans les communications multicast doit tenir compte des facteurs de groupe pouvant influencer le mécanisme de sécurité à mettre en place. Ces facteurs sont de deux ordres :

- les facteurs liés aux caractéristiques du groupe;
- les facteurs relevant des besoins de sécurité du groupe.

1.7.1 Facteurs liés aux caractéristiques du groupe

Certains paramètres affectent de façon cruciale le type d'architecture de sécurité devant être utilisé :

- La taille du groupe,
- les caractéristiques des membres, dont la vitesse du système, les protocoles supportés,
- la nature de la dynamique au sein du groupe (i.e statique ou avec ajouts et départs) et la fréquence des changements au sein du groupe,
- le type de contrôle de groupe : centralisé ou non,
- la durée de vie du groupe,
- le type de groupe : ouvert ou fermé,
- le type d'application : 1 à N ou N à M,
- le volume et le type de trafic.

1.7.2 Facteurs liés aux besoins de sécurité

Dépendamment de la nature du groupe, les besoins en matière de sécurité varient, mais on distingue les besoins de sécurité de base suivants :

1.7.2.1 Le contrôle d'accès

Il est basé sur deux systèmes :

- Un système de gestion de la politique du groupe définissant la politique d'accès au groupe dont le mode d'adhésion, les droits d'accès.
- Un système d'autorisation des membres qui traite les requêtes d'adhésion, authentifie le droit d'accès et autorise l'adhésion.

1.7.2.2 La confidentialité

Elle peut être de deux ordres :

- À court terme ou **éphémère** et consiste à empêcher les non membres à avoir accès aux informations transmises,
- À long terme et consiste à protéger les données pour une longue période de temps. [30]

1.7.2.3 L'authentification

Elle peut s'appliquer de deux façons :

- L'authentification de groupe qui permet à chaque membre de reconnaître que le message reçu provient d'un membre du groupe.
- L'authentification de la source, permettant d'identifier de façon spécifique la source du message.

1.8 Les problèmes de sécurité et leurs solutions

La notion de groupe multicast permet de dégager trois principales propriétés :

- Lors des communications de groupe, tous les membres du groupe reçoivent tous les paquets envoyés à l'adresse du groupe;
- Le multicast permet au groupe d'être ouvert, c'est-à-dire d'être transparent à l'égard de la source;
- Lors de communications de groupe, chaque hôte peut envoyer les données au groupe sans toutefois que la source du message soit authentifiée.

Les propriétés ci-dessus induisent trois problématiques de sécurité, les vulnérabilités qu'elles occasionnent. Le tableau ci-dessous (tableau 2) récapitule les problèmes de sécurité liés à chaque propriété, les failles qu'elle occasionne et les solutions à apporter [7].

Propriétés	Groupe ouvert	Non spécification du destinataire	Non spécification de la source
Problèmes	Accès ouvert au contenu distribué	Non individualisation des données reçues	Transmission ou émission libre de données
Vulnérabilités	Déni de service *Écoute clandestine*	Vol de service	Déni de service *Mascarade*
Solutions	Contrôle d'accès du récepteur *Gestion de la clé de groupe*	Authentification du récepteur par marquage	Contrôle d'accès de la source *Authentification de la source*

Tableau 2 : Problèmes de sécurité et solutions

1.9 Conclusion

Les éléments présentés dans ce chapitre constituent les principes fondamentaux du multicast tel qu'il est définit pour tous les types de réseaux, en particulier les réseaux filaires, et tous les types de communications de groupe. Il existe cependant des spécificités propres à chaque réseau. La section qui suit introduit les réseaux sans fil ad hoc et le multicast tel qu'il y apparaît. La suite de ce mémoire se répartit comme suit : dans le chapitre 2, je fais une présentation des réseaux sans fil ad hoc et des différentes problématiques qui leurs sont propres. Dans le chapitre 3, il sera question de la présentation des différentes méthodes de distribution de clés qui s'appliquent lors des événements de groupe à l'intérieur d'un groupe multicast. Au chapitre 4, je décris ma méthode qui consiste à proposer une méthode de distribution de clés de cryptage basée sur la prise en compte de la mobilité des nœuds et de la dynamique des événements de groupe lors de communications multicast ; au chapitre 5, je décris la méthodologie des simulations et au chapitre 6, je présente les résultats obtenus.

CHAPITRE 2

Les réseaux mobiles ad hoc

2.1 Introduction

Il y a longtemps que la radio et la télévision sont entrés dans les foyers des gens et les transportent instantanément partout dans le monde. Comme le téléphone, ce sont des moyens puissants de communication. De nos jours, les antennes paraboliques poussent sur les toits des édifices comme de véritables champignons, la vente des téléphones cellulaires et d'autres appareils sans fil a explosé; preuve que les communications sans fil sont en pleine expansion. Même les régions autrefois dites enclavées sont aujourd'hui accessibles grâce à la magie des ondes. Mais il n'est pas possible de couvrir tous les espaces et certaines régions resteront difficiles d'accès. Les réseaux ad hoc apparaissent peut être comme une solution de réseautage pour les zones difficilement atteignables. Déjà, des interventions militaires dans les zones risquées se fondent sur la technologie des réseaux ad hoc pour la coordination de leurs opérations. Nous présenterons dans ce chapitre, la structure, le fonctionnement et les avantages des réseaux ad hoc.

2.2 Définition des réseaux ad hoc

Les communications sans fil sont une technologie de plus en plus populaire. Les réseaux sans fil sont ceux dans lesquels les infrastructures ne sont pas reliées par des câbles. Les supports de transmission sont généralement les ondes radios, les ondes électromagnétiques, les micros ondes, les ondes infra rouges. Certains d'entre eux, en plus d'offrir la technologie sans fil, supportent également la mobilité. On distingue en définitive deux types de réseaux sans fil : les réseaux à infrastructures fixes et ceux sans infrastructures qu'on appel ad hoc.

Les réseaux mobiles ad hoc sont des réseaux (ou ensembles) de nœuds mobiles sans fil, sans infrastructure existante ni administration centralisée, formés dynamiquement. La communication se fait par sauts multiples d'une source vers une destination, c'est-à-dire que l'information est acheminée d'un nœud intermédiaire à l'autre de la source jusqu'à la destination. Chaque nœud peut aussi communiquer directement avec ceux situés dans sa portée de transmission [34]. Dans les réseaux ad hoc, chaque nœud se comporte comme un routeur.

2.3 Fonctionnement des réseaux ad hoc

Les ondes radio sont le principal support de transmission dans les réseaux ad hoc. La communication entre périphériques (mobiles) se fait de deux façons : directe ou par routage [32].

La communication est directe lorsque, sans aucune perturbation, les nœuds suffisamment proches peuvent s'envoyer directement des données (Figure 1).

La communication par routage (ou par saut) s'effectue entre deux nœuds relativement éloignés (hors de portée), les données étant envoyées par l'entremise de nœuds intermédiaires qui vont jouer le rôle de routeur (Figure 2).

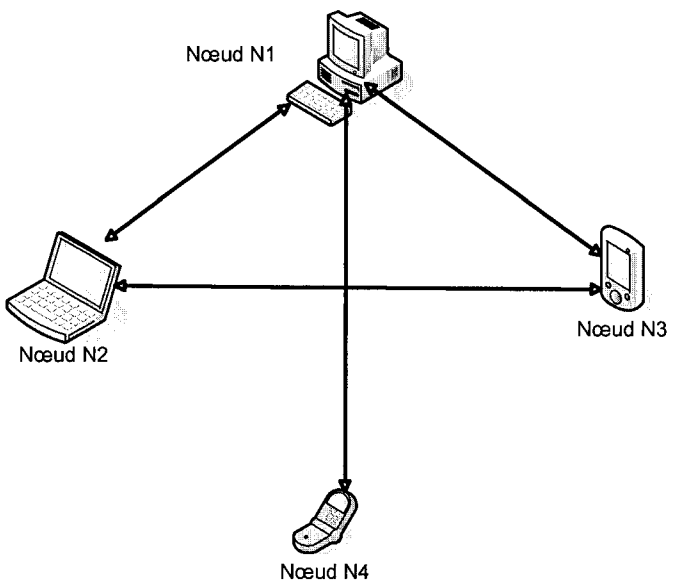

Figure 1 : Communication directe entre deux nœuds.

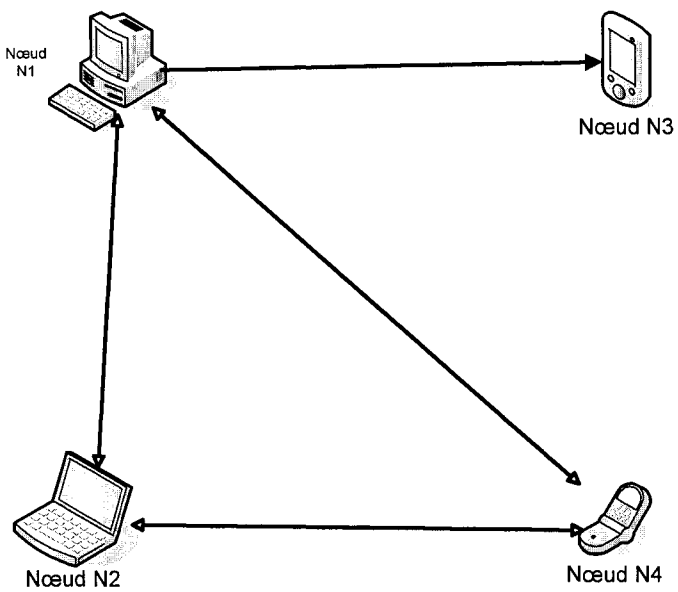

Figure 2 : Communication par routage entre N4 et N3.

2.4 Intérêt et caractéristiques des réseaux ad hoc

L'importance des réseaux filaires et celui des réseaux sans fil traditionnels ne sont plus à démontrer, il existe de moins en moins de zone isolée du reste du monde. Il y a cependant encore des zones où il serait extrêmement coûteux et périlleux d'implanter des infrastructures de communication. Il y a aussi des opérations critiques où il est plus avantageux de disposer d'un réseau autonome et fiable; et c'est dans cette optique que les réseaux ad hoc trouvent pleinement leur sens. De façon générale, on distingue deux types de réseaux mobiles ad hoc :

- **Le MANET isolé** : le réseau mobile sans fil ad hoc (Mobile Ad hoc NETwork ou MANET) est isolé lorsqu'il est autonome et constitué de nœuds sans fil mobiles situés dans une même zone géographique, les noeuds échangent l'information par utilisation d'ondes radio et d'un protocole de routage ad hoc [30].

- **Le MANET connecté à un réseau externe** : le MANET est connecté à un autre réseau par l'intermédiaire des passerelles. Des nœuds frontaliers du Manet sont connectés au réseau externe grâce à des passerelles [30].

2.4.1 Caractéristiques des réseaux mobiles ad hoc

Les réseaux mobiles ad hoc présentent les caractéristiques suivantes :

- Ils ont des ressources limitées (CPU, mémoire, support de stockage, bande passante, batterie).
- Ils présentent un changement constant de topologie.
- La portée de la connectivité entre les nœuds du réseau est restreinte.
- Ce sont des réseaux spontanés.
- Ils sont géographiquement déployés dans un rayon limité.
- Ils sont généralement de petite taille.
- La connectivité est très variable et changeante à cause de la connexion et de la déconnexion constante des nœuds.

2.4.2 Intérêt des réseaux mobiles ad hoc

En comparaison avec les réseaux filaires et ceux sans fil à infrastructures fixes, les réseaux ad hoc offrent certains avantages à cause des éléments qui suivent :

- L'absence d'infrastructures fixes, coûteuses et difficiles à déployer et à maintenir.
- Le manque de câblage.
- Les réseaux ad hoc peuvent facilement être déployés.
- Ils permettent la mobilité des nœuds et donc les mouvements de groupe.
- Ils sont facilement extensibles.
- Ils peuvent constituer une extension des réseaux à infrastructure fixe, principalement dans les zones ou les infrastructures de réseau sont impossibles ou difficiles à implanter [32].

2.5 Applications des réseaux ad hoc

Les réseaux ad hoc peuvent servir dans les environnements difficiles d'accès et comme prolongement aux différents réseaux à infrastructures. C'est avant tout dans les opérations militaires que ces types de réseaux se sont révélés, mais depuis quelques années, les réseaux ad hoc peuvent être déployés pour différents types d'applications dont voici quelques unes présentées au tableau 3 [34] .

Applications	**Description/service**
Réseaux tactiques	Communications et opérations militaires.
Réseaux de capteurs	Applications environnementales (ex : repérage des animaux)
Services d'urgence	Recherches et opérations de sauvetage.
Commerce	Commerce électronique (ex : paiement de facture de n'importe où).
Éducation	Conférence, réunion, salle de classe ou de conférence virtuelle.
Jeux	Jeu en ligne multi usagers
Renseignements sur la localisation	Services de repérage (ex : transfert automatique d'appel).

Tableau 3 : les applications des réseaux ad hoc

Ils peuvent également être utilisés :

- Comme équipement de campus.
- Dans le domaine de la santé.
- Lors des catastrophes naturelles.
- Pour les travailleurs mobiles.
- Pour augmenter la couverture radio dans les endroits suffisamment denses.

2.6 La sécurité dans les réseaux ad hoc

Sécuriser les réseaux ad hoc est une tache particulièrement difficile à cause de [33] :

- La vulnérabilité des canaux : les messages peuvent être écoutés ou injectés dans le réseau.
- La vulnérabilité des nœuds : les nœuds du réseau sont facilement accessibles par les intrus.

- Le manque d'infrastructure : les solutions classiques basées sur la présence d'autorité de certification ou de serveur centraux sont inapplicables ou difficilement applicables.
- Le changement de topologie : La mobilité des nœuds entraîne un changement fréquent de topologie qui résulte en un changement de route, des partitionnements (séparations) fréquents et des pertes de paquets possibles [34].

2.7 Le multicast dans les réseaux ad hoc

Les caractéristiques du multicast tel qu'elles ont été présentées au chapitre précédent s'appliquent également aux communications de groupe dans les réseaux ad hoc. Cependant, dans les réseaux ad hoc, chaque nœud du réseau est un routeur et participe, si cela est nécessaire, au transfert des messages. Ceci augmente les risques de compromission de la sécurité des communications, mais aussi, à cause des ressources limitées des différents nœuds, certains pourraient se soustraire à la tâche de routage particulièrement gourmande en énergie. La transmission des messages, et donc celle des clés de cryptage demande que soit, par exemple, préalablement établie, une entente de coopération entre différents nœuds, voir même, tous les nœuds du réseau.

Les algorithmes de gestion de clés multicast dans les communications filaires ne satisfont pas aux exigences du sans fil, aussi faut-il les développer afin qu'ils s'accommodent à l'environnement des réseaux sans fil ?

Les paragraphes qui suivent présentent différents algorithmes utilisés dans la gestion des clés multicast. De l'étude comparative qui en découlera, nous tenterons de développer une approche efficace de distribution de clés qui réponde aux exigences de la mobilité et de la limitation des ressources des réseaux ad hoc.

2.7.1 Comparaison entre multicast mobile sans fil et filaire

L'introduction du multicast dans les réseaux sans fil est une avenue intéressante même si elle présente de nombreux défis, surtout dans un déploiement à large échelle. La croissance rapide des technologies sans fil, notamment dans le domaine des applications de masse, la limitation des ressources disponibles prédisposent le multicast comme étant la technologie idéale au support des communications de groupe. Cependant, les solutions proposées pour le multicast dans les réseaux filaires ne s'appliquent pas in extenso dans les réseaux sans fil. Le tableau ci-dessous nous montre les différences existantes entre le multicast filaire et sans fil.

Paramètres	Multicast filaire	Multicast sans fil
Type de lien	Symétriques et fixes	Asymétrique et unidirectionnelle
Bande passante	Complète	Limité et variable
Topologie	fixe	Fixe ou dynamique
Perte de paquets	Très peu (<1%)	Fréquents (1 à 30%)
Changement dans la structure du groupe	Lors des événement de groupe	Lors des événements de groupe et du changement de lieu
Routage	Fixe durant la session	Dynamique à cause de la mobilité
Sécurité	Peu complexe	Très complexe à assurer
Qualité de service	RSVP pour les routes fixes	Inadéquation de RSVP
Fiabilité	Utilisation possible de protocoles de la couche transport	Difficile à cause de la nature des médias et de la mobilité

Tableau 4 : Comparaison qualitative entre le multicast filaire et sans fil.

2.8 Mobilité et Connectivité

Lors de la communication dans un réseau mobile ad hoc, les protocoles de routage établissent un chemin à travers lequel les paquets sont acheminés, mais le mouvement

des nœuds brise ces liens. La mobilité des nœuds a donc pour effet d'interrompre la communication entre nœuds et par ricochet, la livraison des paquets. Le lieu, la durée et la fréquence de ces interruptions varient avec les caractéristiques du mouvement des nœuds. Pandey et Zappala [33], Bai, Sadogopan et Helmy [35] proposent un certain nombre de paramètres permettant de mesurer l'effet de la mobilité sur la connectivité des nœuds et le routage :

- Le nombre de changement de liens : il définit le nombre de changements de liaisons durant la simulation. Un changement de liaison exprimant l'événement par lequel deux nœuds précédemment hors de portée l'un de l'autre se trouvent subitement dans un rayon de portée de communication.
- L'«atteignabilité» : elle définit le nombre de nœuds joignables par sauts multiples (ou transfert) à travers le réseau ad hoc.
- La densité du voisinage : exprime le nombre de nœuds situés dans la portée radio d'un nœud donné.
- La dépendance spatiale : elle caractérise le degré avec lequel deux nœuds se déplacent dans la même direction et à la même vitesse.

Ils ont établi les relations mathématiques suivantes permettant d'exprimer les paramètres définit ci-dessus :

R : définit le rayon de portée radio

$v_i(t)$: la vitesse en mètre par seconde du nœud i à l'instant t

$D_{i,j}(t)$: la distance en mètre entre les nœuds i et j à l'instant t

$RD = \frac{vi(t).vj(t)}{|vi(t)*vj(t)|}$, la relation directionnelle entre les nœuds i et j à l'instant t

$RV = \frac{\min(\ vi(t), vj(t)\ )}{\max(\ vi(t), vj(t)\ )}$, le ratio de la vitesse des nœuds i et j à l'instant t

$P_{i,j} = \left\{ \begin{matrix} 1, \text{ si } Di,j(t) < R \\ 0, \text{ si } Di,j(t) > R \end{matrix} \right\}$, la probabilité que deux nœuds i et j soient à l'intérieur de la portée radio l'un de l'autre.

La densité du voisinage du nœud i est donnée par la relation : $N_i = \sum_{j=0}^{N} Pi,j$ ou N est le nombre de nœuds du réseau.

La dépendance spatiale des nœuds i et j à l'instant t est donnée par :

D_s (i, j, t) = RD * RV avec D_s (i, j, t) = 0 si $D_{i,j}$ (t) = C * R (C>1) où C est une constante entière.

La «joignabilité» d'un nœud est optimale si sa densité du voisinage est de N-1 où N est le nombre total de nœuds dans le réseau.

2.9 Conclusion

Les réseaux sans fil connaissent une croissance phénoménale, ils sont plus conviviaux car ils offrent une plus grande latitude de communiquer puisqu'ils libèrent des contraintes de l'espace. Mais on observe également des problèmes importants au niveau de la qualité des services, de l'utilisation de la bande passante et de la sécurité. La mobilité est un facteur clé au niveau de la perte de paquets et du partitionnement du réseau. La déconnexion et la fragmentation au sein du réseau, dues entre autre à la mobilité des nœuds, pourraient être deux phénomènes très importants lors des communications de groupe dans les réseaux mobiles sans fil ad hoc; dans ce cas, ils pourraient affecter le processus de distribution de clés de groupe et donc de la communication multicast.

Le chapitre suivant présente les différentes méthodes de distribution ou d'établissement de clés de groupe lors de communication multipoints dans les réseaux filaires et sans fil. Une étude comparative de ces méthodes présente les avantages et les inconvénients de chacune d'elle.

CHAPITRE 3

Méthodes d'établissement des clés de cryptage lors des communications multicast

3.1 Introduction

Lors de la communication entre deux systèmes informatiques, il y a échange de paquets. Mais les données transmises sur un réseau public comme Internet peuvent facilement être interceptées et analysées pour en extraire des informations sensibles. Ce genre de pratiques et les autres attaques visant à compromettre les systèmes et les institutions qui les abritent sont de plus en plus courantes. Pour éviter de tels désagréments et fiabiliser les communications, des mesures de sécurité sont mises en place, parmi lesquelles le chiffrement des données à l'aide d'une clé de cryptage. Les entités impliquées dans la communication doivent au préalable adopter un protocole de communication sûr grâce auquel ils pourront s'échanger, de façon sécuritaire, les clés de chiffrement qui seront utilisées. Cet ensemble de règles régissant la communication entre deux entités doit garantir entre autre l'**authenticité** des parties et l'**intégrité** des données transmises. Dans les communications point à point, les entités qui communiquent s'échangent généralement leurs clés, partagées ou publiques, dépendamment de la méthode cryptographique adoptée. On parle alors d'un échange de clés point à point. Cependant, lors des communications de groupe, un serveur central est souvent chargé de distribuer les clés de cryptage aux différents membres du groupe multicast, on parle alors de la distribution des clés de cryptage. Quoiqu'il en soit, qu'il s'agisse d'un échange de clés ou d'une distribution, le processus de cryptage dans son ensemble est une opération gourmande en temps CPU et en utilisation de la bande passante. C'est pourquoi, les algorithmes de distribution des clés ont le souci de réduire la consommation des ressources. Les pages qui suivent présentent plusieurs approches de distribution de clés de cryptage.

3.2 Les approches de distribution de clés

3.2.1 GÉNÉRALITÉS

Lors des communications de groupe, les participants doivent posséder la clé de groupe pour pouvoir décrypter les messages reçus en multicast. Cette clé et d'autres clés privées leurs sont remises par le ou les contrôleurs de groupes à travers un canal sécurisé. Plusieurs méthodes de distributions ont été développées et peuvent être classées dans deux grandes catégories : On les subdivisent en méthodes contributives et méthodes distributives. Comme nous allons les détailler plus loin dans ce chapitre, les méthodes contributives peuvent être totales ou partielles, les méthodes distributives, centralisées ou décentralisées. La figure ci-dessous (Figure 3) présente les différentes catégories suivant lesquelles on peut classer les méthodes de distribution de clés de cryptage lors des communications de groupe.

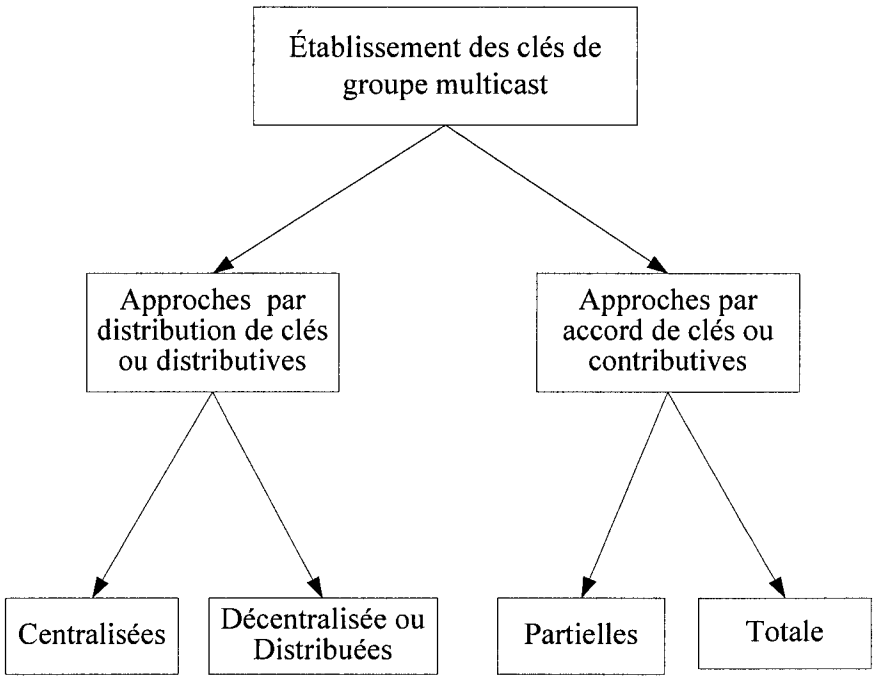

Figure 3: Classification des approches d'établissement de clés de groupes multicast

3.2.2 Critères d'évaluation et du choix du protocole de distribution

Dans [16] et [34], les auteurs présentent un certain nombre de critères à appliquer lors d'une communication de groupe pour évaluer et choisir le protocole de distribution de clés. Ils citent les paramètres suivants :

- Le nombre d'itérations ou rounds effectué lors de la génération de la clé de groupe.
- Le nombre de messages échangés entre les membres dans le processus de création de la clé de groupe.
- Le traitement (calculs effectués) au moment de l'initialisation du groupe.
- L'utilisation de l'algorithme de Diffie-Hellman (DH) dans la génération de la clé.

Par ailleurs, Hietalahti [29] ajoute que la conception et le développement d'un protocole efficace de distribution ou d'établissement de clés lors de communication de groupe doivent tenir compte:

- Du type de méthode de distribution que l'on veut adopter.
- Des caractéristiques du réseau.
- De la nature des informations transmises.
- Des avantages et inconvénients des approches déjà éprouvées.

3.2.3 Architectures et critères de comparaison

Les architectures de gestion de clés doivent satisfaire quatre principaux critères : la réduction du **temps de création du groupe**, la réduction du **nombre de transmissions**, la réduction de **quantité de mémoire utilisée** et de **l'espace de stockage** des clés. Dans [7], Ammar et Judge nous proposent une classification des schémas de gestion des clefs multicast. Cette classification peut être représentée par le tableau ci-dessous (tableau 4).

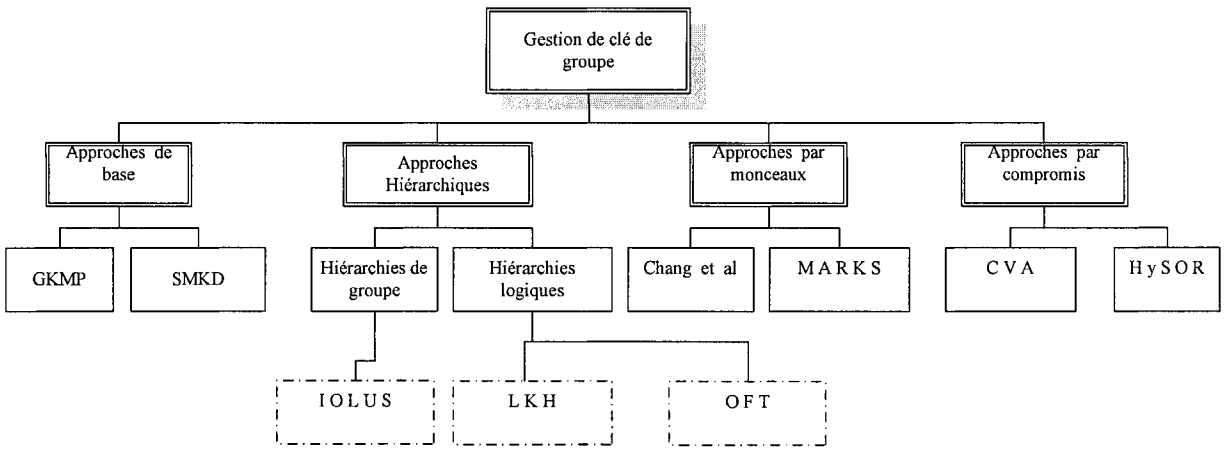

Figure 4: Quelques approches de gestion de clés de multicast.

3.2.4 Opérations dans le groupe

3.2.4.1 Les événements de groupe

Lors des communications multicast, plusieurs types d'événements se produisent au sein du groupe. Ce sont ces événements qui dictent le comportement des différents protocoles qui sous tendent les communications de groupe. Ces éléments sont :

- L'ajout Simple : un ajout est dit simple lorsqu'un seul nouveau participant se joint au groupe multicast.
- L'ajout Multiple : l'ajout est multiple lorsque plusieurs participants se joignent en même temps au groupe.
- Départ Simple : le départ est simple lorsque un seul participant quitte le groupe multicast.
- Départ Multiple : le départ multiple survient lorsque plusieurs participants quittent le groupe à la fois.
- Séparation : il y a séparation lorsqu'un groupe se scinde en deux ou plusieurs groupes.
- Fusion : la fusion survient lorsque deux ou plusieurs groupes indépendants se joignent pour ne former qu'un seul grand groupe.

- Pertes de clés : on parle de perte de clé lorsqu'un participant ne reçoit pas les messages contenant les clés lui permettant de décoder les messages chiffrés suivants. Seulement quelques protocoles en font allusion.

Il faut noter que certains événements de groupe peuvent être volontaires ou accidentels. Dans le cas des départs, des participants peuvent volontairement quitter le groupe le groupe ou en être exclus. Les fusions et les séparations peuvent également être volontaires ou arriver à cause d'un incident du réseau. Certains protocoles traitent différemment les deux aspects, mais nous les inclurons dans la même opération.

Dans le cas des réseaux mobiles, on peut ajouter la perte de connectivité qui est un phénomène généralement fréquent dans ce type de réseau. Elle peut être due à une défaillance subit de l'énergie du périphérique ou à un transfert intercellulaire lors du déplacement du noeud.

3.2.4.2 Sécurité des opérations de groupe

De façon générale, les communications multicast sécuritaires doivent satisfaire un certain nombre de contraintes de sécurité : authentification, confidentialité, intégrité, non-répudiation et l'indépendance des clés.

- L'authentification : elle sert à authentifier l'identité de l'entité impliquée dans la communication, donc à s'assurer qu'elle est la personne ou le système qu'elle prétend être. L'authentification peut être vérifiée par l'utilisation de mécanismes comme signature numérique.

- La confidentialité : de façon générale, elle suppose que seul les détenteurs des clés de chiffrement et de déchiffrement doivent pouvoir crypter et décrypter les paquets transmis et ainsi avoirs accès à l'information contenue dans les données transmises. Dans le cas particulier des communications de groupe, on peut

distinguer comme conséquence aux opérations de groupe, deux autres formes de confidentialité :

- La confidentialité du passé : protéger les données transmises avant l'arrivée d'un nouveau membre pour qu'il ne puisse jamais y avoir accès.
- La confidentialité du futur : protéger les données transmises après le départ d'un membre pour qu'il ne puisse plus continuer à les recevoir.

- Indépendance de la clé : tout adversaire disposant d'un sous ensemble de clés ne peut découvrir aucune clé de groupe (passée ou à venir).
- La résistance à la collusion : elle vise à empêcher que des anciens membres du groupe, en échangeant leurs informations secrètes, puissent avoir accès aux données transmises.
- La résistance à la compromission : Elle consiste à assurer l'inviolabilité de la politique de sécurité existante [34].

La gestion des clés multicast peut s'avérer coûteuse en temps de calcul, en espace de stockage, en bande passante, en consommation d'énergie, surtout pour les périphériques sans fil. D'où la nécessité de prendre en compte les facteurs suivants lors du développement d'une architecture de gestion de clés : le niveau de sécurité, le coût, l'initialisation du système, la politique de sécurité, les procédures de contrôle d'accès, la performance et les mécanismes de soutien [4].

3.3 Protocoles par distribution de clés

Dans ces architectures, un ou des contrôleurs de groupe sont chargés de générer et distribuer la clé de groupe. On distingue deux classes : les approches dites centralisées et les approches dites décentralisées (ou distribuées) [42].

3.3.1 Les approches centralisées

Dans les approches centralisées, une seule entité est responsable de la génération et de la distribution des clés aux membres du groupe. La clé privée de chaque membre lui est envoyée à travers un canal sécurisé par le gestionnaire du groupe. La clé de

groupe quant à elle est envoyée à chaque membre dans un message chiffré avec sa clé individuelle. La distribution centralisée peut être simple ou en arbre et cherche à minimiser la consommation du CPU, de l'espace de stockage des clés et de la bande passante. L'efficacité du protocole se mesure ici en fonction de la taille des messages, des clés, des ressources disponibles, de la confidentialité passée et future et de la collusion [35].

De façon générale, les approches centralisées sont surtout efficaces dans les communications de un-à-plusieurs. Leur principal faiblesse réside dans le fait que :

- Le serveur central constitue le seul point d'échec et la cible d'attaques.
- Le fonctionnement est limité en cas de partitionnement du réseau [15].
- Le serveur central doit être disponible en tout temps même en cas de partitionnement du groupe [34].

3.3.2 Les approches centralisées à un niveau ou simples

Ces approches sont basées sur les méthodes de chiffrement des communications unicast. Une entité centrale, le KDC (Key Distribution Center), encrypte la clé de groupe avec la clé privée de chaque membre et la lui envoie. Elle effectue donc un nombre d'envoie égal au nombre de participants de la session multicast. À la suite de cette opération, chaque membre peut déchiffrer les messages destinés au groupe. Les principaux protocoles de distribution centralisée à un niveau sont : le protocole GKMP (Group Key Management Protocol) [4][5].

3.3.2.1 Le Protocole GKMP (Group Key Management Protocol)

3.3.2.1.1 Création de groupe

Dans cette approche, le contrôleur et le second membre à joindre le groupe créent le paquet de la clé de groupe (GKP) contenant la clé de chiffrement des messages de groupe (GTEK) (ou clé de groupe) et celle de chiffrement de la clé de groupe (GKEK). À chaque ajout, le nouveau membre reçoit une copie du GKP. À chaque renouvellement

de la clé, un nouveau GKP est généré et encrypté avec la GKEK courante. Cependant, à chaque départ, il faut recréer un nouveau groupe dont le membre sortant est exclu. La figure 5 nous montre un modèle d'arbre de distribution dans une approche centralisée à un niveau.

3.3.2.1.2 Avantages

- Cette méthode est largement répandue
- Méthode assez simple à implanter

3.3.2.1.3 Inconvénients :

- Il faut récréer le groupe entier à chaque départ
- Ce protocole ne satisfait pas à la mise à l'échelle et aboutit rapidement à la saturation du gestionnaire du groupe (Group Manager ou GM).

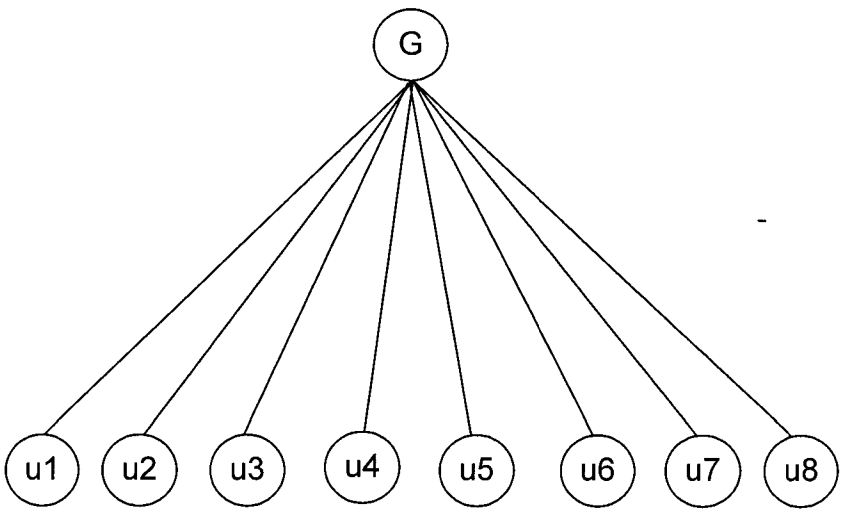

Figure 5 : exemple d'approche centralisée.

3.3.3 Les approches centralisées en arbre

Proposées entre autres par **Wallner et al** [1] et **Caronni et al** [4], dans ces approches :

- Les clés sont réparties sur un arbre (équilibré ou non) dont les feuilles forment les clés de chaque membre (Figure 6).

- Le contrôleur de groupe maintient un arbre de distribution de clés.
- À la racine se trouve la clé de groupe.
- À chaque noeud intermédiaire se trouve une clé de chiffrement de la clé de groupe.
- Chaque membre reçoit une clé de chiffrement de la clé de groupe et les clés de chiffrement de la clé de groupe de chaque nœud parent jusqu'à la racine.
- Chaque membre du groupe stocke toutes les clés sur son chemin depuis la base jusqu'à la racine.
- À l'ajout d'un nouveau membre, on inclut sa clé privée dans l'arbre et toutes les clés des parents jusqu'à la racine doivent être changées afin de préserver la confidentialité du passé et du futur.
- Chaque clef du nœud parent est encryptée avec celle du (ou des) fils immédiats.
- Lors du rafraîchissement de la clé de groupe, le gestionnaire de groupe (GM) utilise la hiérarchie des clefs pour exclure un ou des participants du groupe. Toutes les clés intermédiaires compromises et la clé de groupe sont alors remplacées durant le processus.

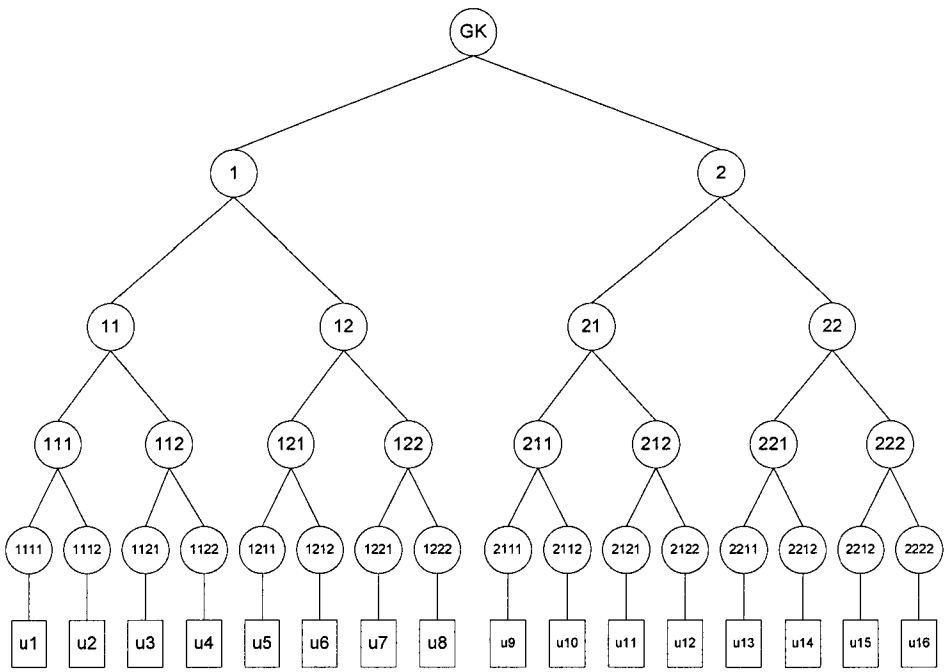

Figure 6 : Structure de l'arbre hiérarchique

3.3.3.1 Avantages :

- Les approches centralisées en arbre minimisent les problèmes liés à la complexité de la distribution de la nouvelle clé sans appel à un tiers parti. Ce qui implique quatre éléments :
    - Complexité de communication : le nombre de messages envoyés est réduit de O(n) à O(log (n)).
    - Complexité de stockage des membres du groupe : le nombre clés stockées est réduit à O(log(n)).
    - Complexité de stockage du gestionnaire de groupe (GM) : O(n)
    - Complexité du temps de calcul : O(1).[9]

3.3.3.2 Le protocole Logical Key Hierarchy++ ou LKH++

Ce protocole [43] est une extension de la méthode en arbre LKH initialement développé pour la gestion des clés dans les communications multicast filaires, il a été adapté pour les réseaux sans fil. Il utilise une méthode cryptographique symétrique pour assurer une faible consommation des ressources : le temps de calcul, l'échange des messages, l'espace mémoire, la consommation d'énergie. Dans cette approche, chaque membre peut soit générer localement les clés, soit les recevoir du serveur central. La clé de chaque nœud intermédiaire est générée en utilisant la clé secrète du fils de gauche en lui appliquant une fonction de hachage. Elle est ensuite distribuée à tous les autres membres qui l'utilisent. La figure 7 illustre une phase d'initialisation de la clé de groupe ou les P_i représentent les clés secrètes de chaque membre U_i, et les K_i, les clés intermédiaires obtenues par application de la fonction de hachage H sur la clé du fils de gauche. Dans le cas de la génération par le contrôleur, elle se fait à partir de sa représentation interne de l'arbre des clés.

3.3.3.2.1 Avantages

Ce protocole vise avant tout à améliorer la création du groupe, ce qui par conséquent permet d'améliorer les opérations d'ajout et de départ des membres.

- Le nombre de clés stockées par chaque membre est une fonction logarithmique du nombre de membres du groupe.
- Le protocole utilise un chiffrement symétrique qui réduit la taille des clés, les efforts de calcul lors de l'encryptage et du décryptage et conserve l'énergie des périphériques.
- Le nombre de messages multicast reçus du contrôleur est réduit.

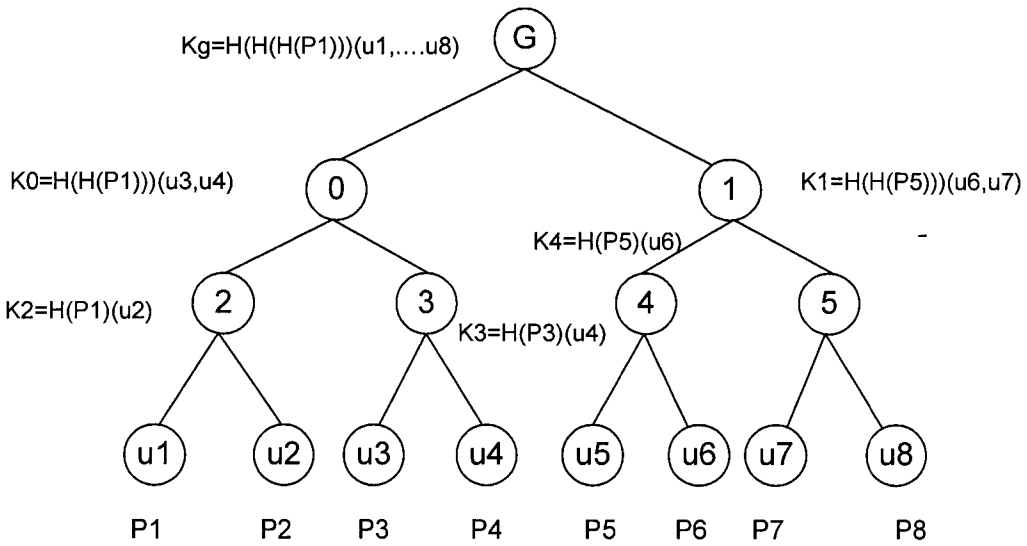

Figure 7 : Arbre d'initialisation de LKH++

3.3.3.3 L'arbre de fonction à un sens OFT (One way Function Tree)

La technique OFT [3] est une amélioration de l'arbre binaire hiérarchique proposée par McGrew et Sherman [44]:

- Chaque nœud est associé à deux clés de cryptage : une clé secrète K_x et une clé cachée K'_x obtenue par application d'une fonction à un sens sur K_x.

- La clé cachée est obtenue à partir d'une fonction à sens unique g selon la formule K '_x = g(K_x).
- Les clés secrètes sont transférées à chaque membre par le contrôleur à travers un canal sécurisé.
- Les clés des nœuds intermédiaires sont obtenues par mixage à partir des clés cachées de ses deux fils selon la formule K_x = f(g(K_g),g(K_d)).
- Chaque membre possède ses clefs de chiffrement et les clefs cachées de tous les nœuds jumeaux de ses parents jusqu'à la racine.

3.3.4 Les approches décentralisées

La gestion décentralisée du groupe multicast implique que sa gestion est confiée à plusieurs entités. Chaque méthode a cependant sa propre technique ; mais on rencontre les constantes suivantes qui caractérisent l'ensemble des approches décentralisées :

- Dans les approches décentralisées, le groupe est divisé en sous-groupes et la gestion est soit confiée à plusieurs contrôleurs de sous-groupes [1][35], soit à un contrôleur sélectionné dynamiquement parmi les membres [15].
- Les contrôleurs de sous-groupes sont néanmoins administrés par un contrôleur de groupe dont ils sont plus ou moins dépendants.

3.3.4.1 Avantages :

- La gestion distribuée est robuste et s'utilise bien dans les communications de plusieurs à plusieurs et elle reste stable en cas de fractionnement du réseau.

3.3.4.2 Inconvénients :

- Elle présente un handicap majeur car à chaque changement de la clé du groupe. Le serveur ou le contrôleur doit rétablir et maintenir des canaux sécurisés pour distribuer les nouvelles clés aux membres, ce qui consomme la bande passante et nécessite l'utilisation du CPU. Ce qui fait diminuer l'énergie des batteries dans le cas des réseaux sans fil.

L'efficacité des protocoles peut être mesurée en fonction des attributs suivants : l'indépendance des clés, le degré d'indépendance des contrôleurs de sous-groupes à l'égard d'un contrôleur central, l'influence du renouvellement local des clés sur le groupe (1 affecte n), le degré de dépendance entre le chemin de données et celui des clés, le type de communication (1 à n ou n à n) [35].

3.3.5 Exemples d'approches décentralisées

3.3.5.1 Le protocole Iolus

Ce protocole [46] propose la subdivision du groupe multicast en sous-groupes indépendants gérés par des contrôleurs de sous-groupes appelés des GSA (Group Security Agents). Les GSA peuvent être le GSC (Group Security Controler), responsable de la gestion de tout le groupe ou des GSI (Group Security Intermediaries), qui sont des proxy groupés par niveau dans une structure de distribution hiérarchique et responsables de la transmission des messages et des clés au sein de leur sous-groupe mais aussi aux autres participants d'autres sous-groupes. Pour le faire, ils connaissent les clés des sous-groupes auxquels ils sont reliés et peuvent utiliser ces dernières pour chiffrer les messages qui leur sont destinés (voir figure 8).

3.3.5.1.1 Création du groupe

Le groupe est crée par le GSC qui définit les politiques de sécurité. Une fois ces politiques mises en place, les GSI et les autres membres de son sous-groupe sont admis sur la base des contraintes de sécurité en vigueur. Les membres désireux de se joindre au groupe peuvent s'inscrire auprès des GSI.

3.3.5.1.2 Les ajouts simples

Les ajouts et départs se font au niveau du sous-groupe sans que l'entité centrale intervienne. Lors de l'ajout, le GSA change la clé de sous-groupe, l'encrypte avec la clé actuelle du sous-groupe et la diffuse aux membres déjà présents dans le sous groupe. Il

utilise ensuite un tunnel sécurisé pour envoyer la nouvelle clé de sous-groupe au nouveau membre, encryptée avec sa clé privée.

3.3.5.1.3 Les départs simples

Lors du départ d'un participant, le GSA envoie un seul message multicast contenant la clé de sous-groupe chiffrée avec la clé privée de chacun des membres du sous-groupe.

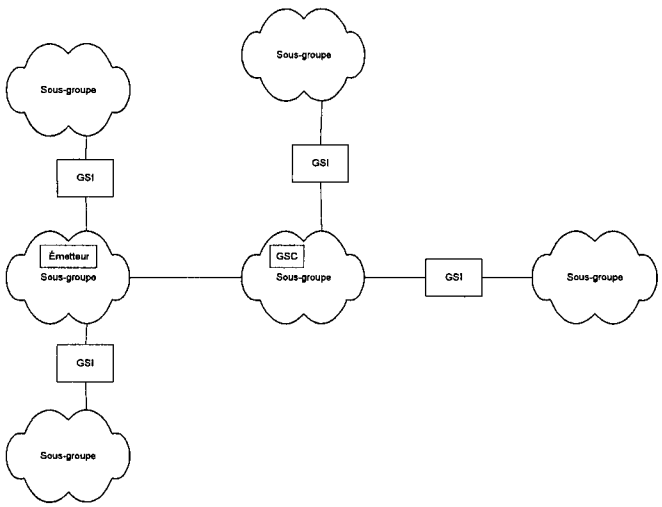

Figure 8 : Division du groupe avec la méthode Iolus

En conclusion pour acheminer des paquets cryptés dans un sous-groupe ou vers un autre sous-groupe, les moyens suivants sont envisageables : le GSI peut utiliser un tunnel sécurisé, il peut décrypter le message reçu des autres GSI, le chiffrer avec sa clé de sous-groupe avant de l'envoyer aux participants. Ceci est par conséquent coûteux en temps de chiffrement.

3.3.5.1.4 Avantages

- L'ajout ou du départ d'un membre affecte seulement la clé du sous groupe auquel il appartient.
- Le regroupement des membres en sous groupe facilite la mise à l'échelle.

3.3.5.1.5 Inconvénient

- Le transfert d'informations entre sous-groupes peut être coûteux en temps de chiffrement.

3.3.6 Approches par contribution

Ce type d'architecture suppose la participation de tous les membres du groupe multicast inscrit à la session en cours. Cette contribution peut s'adresser à tous les membres du groupe, on dit alors qu'elle est totale. Elle peut également s'adresser à certains participants investis de la responsabilité de générer les clés, on dit qu'elle est partielle. Cependant, qu'elles soient totales ou partielles, les approches contributives présentent les caractéristiques suivantes :

- Il n'y a pas de contrôleur central de groupe,
- Chaque membre participe à la génération de la clef de groupe.

3.3.6.1 Avantages

- Ils éliminent les problèmes liés à l'utilisation d'un serveur central [34].
- Ils sont généralement efficaces dans les petits groupes [11].

3.3.6.2 Inconvénients

- Ces approches présentent des lacunes quant à la tolérance aux pannes et de la dynamique du groupe [15].
- Le temps de traitement et les besoins de communication augmentent linéairement avec le nombre de membres [35].
- La liste des membres doit être connues de tous les membres du groupe [35].

3.3.6.3 Valeurs limites des paramètres d'évaluation des protocoles

L'établissement de la clé de groupe dans un protocole par contribution des membres nécessite

- Des échanges de clés.
- L'envoie de messages.
- Un nombre de rounds à l'intérieur duquel la clé de groupe est calculée.

Becker et Wille [20] présentent dans le tableau ci-dessous, le nombre minimum d'échanges, de messages et de rounds nécessaires dans un protocole contributif P pour distribuer l'information, par diffusion ou sans diffusion.

Protocole P	**Paramètres dans les protocoles par contribution**			
	Nombre minimum d'échanges	**Nombre minimum de rounds simples**	**Nombre minimum de rounds synchrones**	**Nombre minimum de messages envoyés**
Avec diffusion	$f_1(P) \geq 2*n-2$	$f_3(P) \geq \lceil \log_2 n \rceil$.	$f_4(P) \geq 1$	$f_2(P) \geq 2*n-4$
Sans diffusion	$f_1(P) \geq n$	$f_3(P) \geq \lceil \log_2 n \rceil$.		$f_2(P) \geq n$

Tableau 5: Valeurs minimales des paramètres de mesures d'efficacité des protocoles contributifs

3.3.6.4 Brève description de l'algorithme de Diffie-Hellman

Devant les problèmes d'échanges de clés rencontrés par les méthodes cryptographiques à clé symétrique, Diffie et Hellman [36] proposent une nouvelle approche cryptographique dite à clé publique basée sur l'utilisation d'une paire de clés. Cette méthode permet seulement l'établissement d'une clé secrète entre deux entités communicantes mais non pas leur authentification. Elle opère comme suit :

- deux entités A et B se mettent d'accord sur deux grands nombres premiers, le générateur **g** et un nombre **p.**
- $n > g$ et $g > 1$.
- n est grand de l'ordre de 512 ou 1024 bits pour que l'échange des clefs soit sécurisé.
- A choisit un nombre aléatoire x et B choisit un nombre aléatoire y

- A calcule $\mathbf{X=g^x \pmod n}$ et l'envoie à B
- B calcule $\mathbf{Y=g^y \pmod n}$ et l'envoie à A
- A calcule $\mathbf{k=Y^x \pmod n}$, B calcule $\mathbf{k'=X^y \pmod n}$
- A et B obtiennent la même clé $K = \mathbf{g^{xy} \pmod n}$

3.3.6.5 Le protocole Hypercube

Le protocole Hypercube [46] [36] est une amélioration du protocole d'Ingemarsson et al. où le nombre de participants est une puissance de 2.

3.3.6.5.1 Création du groupe

- L'initiateur du groupe introduit les nœuds au sein du groupe en les disposant dans un hypercube de dimension d.
- Chaque nœud est l'origine d'un repère de dimension d.

3.3.6.5.2 Initialisation de la clé de groupe

La génération de la clé initiale du groupe s'effectue en plusieurs rounds de la façon suivante :

- Chaque nœud génère un nombre aléatoire
- À chaque round, il établit une clé commune k_i avec un de ses voisins dans une direction i donnée.
- À la fin de tous les rounds, chaque participant possède la clef de groupe.

3.3.6.5.3 Avantages

- Il offre un nombre minimal de rounds ($n=2^d$).
- Le nombre de messages ($n\log_2 n$) est supérieur à l'optimum requis.
- Le nombre d'échanges Diffie-Hellman ($n\log_2 n/2$) est supérieur au minimum.

3.3.6.5.4 Inconvénient

- Il est vulnérable aux multiples pannes dues aux changements de topologie [29].

3.3.5.5.5 Exemple d'étapes d'exécution du protocole Hypercube à 4 participants

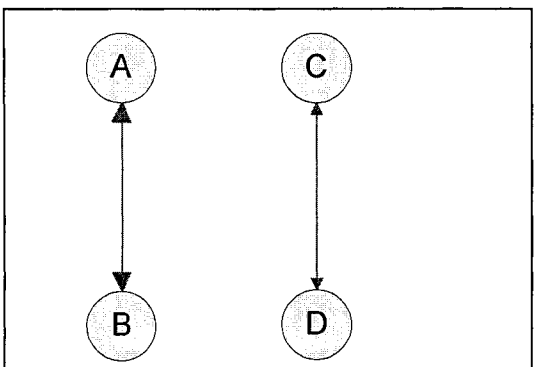

Figure 9: Étape 1 du protocole Hypercube

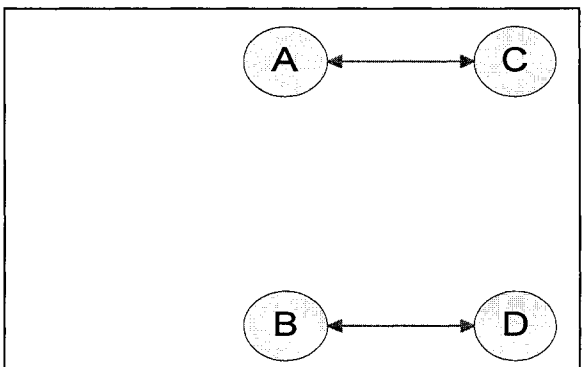

Figure 10: Étape 2 du protocole Hypercube

Étape 1 (Figure 9)

- A et B échangent leurs clés de même que C et D.
- Ils créent les clés $K_{AB} = g^{K_A K_B} \bmod p$ et $K_{CD} = g^{K_C K_D} \bmod p$.

Étape 2 (Figure 10)

- A et C échangent les clés obtenues en à l'étape 1, de même que B et D.
- Ils créent les clés $K_{AC} = g^{K_{AB} K_{CD}} \pmod p = K_{BD} = g^{K_{CD} K_{AB}} \pmod p$.
- Ils obtiennent $K = g^{K_A K_B K_C K_D} \bmod p$.

3.3.6.6 Le protocole Octopus

Ce protocole [15] est une extension et une amélioration du protocole Hypercube qui s'applique à un nombre arbitraire de nœuds.

3.3.6.6.1 Création du groupe

Le groupe est crée au fur et à mesure que les participants s'ajoutent et se structure tel qu'il suit :

- Les quatre premiers participants A, B, C et D à joindre le groupe forment le noyau central du groupe, ce sont les quatre « contrôleurs » du groupe.
- Les autres nœuds se répartissent dans quatre sous ensembles centrés autour A, B, C et D.

3.3.6.6.2 Initialisation du groupe

La génération de la clé initiale du groupe se fait à la suite de plusieurs rounds de la façon suivante :

- À Chaque round i, chaque nœud du noyau central utilise la clé k_{i-1} du round précédent pour établir une clé k_i avec le nœud N_i de son sous-groupe par l'algorithme de Diffie-Hellman.
- Lorsque toutes les clés de sous-groupes ont été établies, les quatre nœuds centraux procèdent à deux rounds supplémentaires et échangent deux à deux leur clé de sous groupe.
- La clé de groupe est ainsi créée et diffusée par le contrôleur aux autres nœuds de son sous-groupe.

3.3.6.6.3 Avantages

- Il minimise, sans utiliser la diffusion, le nombre d'échanges DH, c'est-à-dire le nombre de connexions devant être établies pendant l'exécution du protocole.
- Octopus, tout comme les autres protocoles effectuant les échanges DH, ne requière pas l'établissement d'un canal permanent pour l'échange de clé.

3.3.6.6.4 Inconvénients

- Le nombre de messages dépasse le minimum requis [20].
- Il hérite aussi des vulnérabilités du protocole Hypercube.

3.3.6.6.5 Exemple d'étapes d'exécution du protocole Octopus

Le protocole Octopus à 11 participants s'exécute tel que décrit dans la figure 11 :

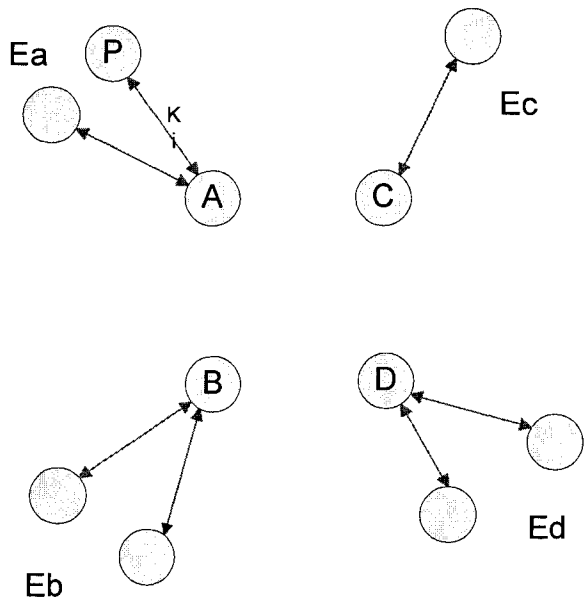

Figure 11: Protocole Octopus à 11 participants

- Chaque participant P des ensembles Ea, Eb, Ec et Ed effectue un échange avec son « contrôleur ».
- Les « contrôleurs » A, B, C et D établissent la clé commune en appliquant le protocole Hypercube à 4 participants.
- La clé est diffusée auprès des participants dans chaque ensemble.

3.3.6.7 Protocole pour la découverte des membres du groupe

Il permet d'établir une séquence dans la contribution des membres du groupe. Lors qu'un nœud veut envoyer sa contribution au nœud suivant, il cherche les membres du groupe et en choisit un en utilisant un protocole de routage ad hoc et l'envoie à l'aide d'un message par inondation. Tout nœud n'ayant pas encore apporté sa participation lui répond. S'il ne reçoit aucune réponse pendant une certaine période de temps, il réessaie. Si aucune réponse n'est encore obtenue, il devient alors le dernier nœud et le contrôleur du groupe. Si par contre il reçoit une réponse, il choisit un nœud comme étant le suivant. Il peut arriver que certains membres soient injoignables, soit parce qu'au moment de la recherche ils participent à la création d'un sous-groupe ou parce qu'ils disposent déjà de

la clé de groupe. Dans un cas comme dans l'autre, il faut les réinsérer à l'intérieur du groupe et leur remettre la nouvelle clé. Cette situation pourrait être fréquente à l'intérieur d'un réseau ad hoc, conduisant ainsi à un perpétuel changement de la clé de groupe [13].

3.3.6.8 Les protocoles de la suite CLIQUES

Ce sont des protocoles de gestion de clefs de groupes qui entrent dans la classe des protocoles qui étendent l'algorithme de Diffie-Hellman [21] au groupe. Ceux abordés ici pour l'établissement de la clef de groupe sont les protocoles GDH.2 et GDH.3.

Les protocoles GDH sont relativement efficaces quant aux opérations de départ et de partitionnement du groupe, mais la fusion requiert autant de rounds que le nombre de nouveaux membres [34].

3.3.6.8.1 Le protocole GDH.2

Il a été adopté pour réduire le nombre de rounds dans le calcul de la clé [21] et se fait en deux étapes : une étape ascendante et une autre descendante par diffusion (Figure 12).

Étape 1 : Elle comporte une phase ascendante de collecte de la contribution des membres dans laquelle, pendant les n-1 rounds, chaque membre Mi envoie à Mi+1 un ensemble de valeurs $\{gS1^{*S2***Si/Sk} / k \in [1,i] \}$ et une valeur cardinale $g^{S_1*S_2***S_i}$ [19] [21].

Étape 2 : Le nœud M_n qui est le contrôleur de groupe, calcule la clé de groupe $K_n = g^{S1*S2***S_n}$ et les dernières valeurs intermédiaires $\{g^{S1*S2***Sn/Si} / i \in [1,n-1]\}$et les diffuse à tous les membres du groupe.

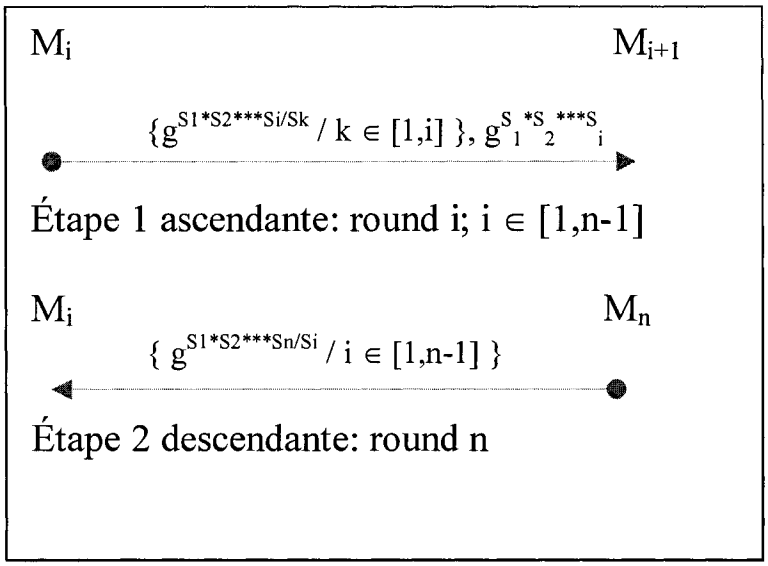

Figure 12 : protocole GDH.2

3.3.6.8.1.1 Ajout simple

- Lors de l'ajout d'un nouveau membre M_{n+1}, le membre M_n génère une nouvelle clé individuelle et calcule une nouvelle clé intermédiaire $k_n = g^{S1*S2***Sn}$ et l'envoie à M_{n+1}.
- M_{n+1} génère sa contribution (clé) et calcule la clé $k_{n+1} = g^{S1*S2***Sn*Sn+1}$.
- M_{n+1} factorise la clé en différentes sous clés $\{gS1^{*S2***Sn/Si} / i \in [1, n-1]\}$ et diffuse l'ensemble aux autres membres du groupe.

3.3.6.8.1.2 Départ simple

- Lors du départ du participant M_p, le contrôleur M_n crée une nouvelle clé individuelle, factorise et supprime la contribution (sa clé) de M_p.
- Il calcule les différentes clés intermédiaires $\{gS1^{*S2***Sn+1/Si} / i \in [1, n]\}$ et les diffuse aux autres membres du groupe.
- Si M_n se retire du groupe, alors M_{n-1} joue le rôle de contrôleur.

3.3.6.8.1.3 Avantage

- Ce protocole réduit le nombre de rounds lors du calcul de la clé de groupe.

3.3.6.8.1.4 Inconvénients

- Le calcul de la clé de groupe est une fonction linéaire du temps.
- Le nombre d'exponentiations est par conséquent élevé.

3.3.6.8.2 Le protocole GDH.3

Le protocole GDH.3 [46] a été proposé pour réduire les coûts liés aux opérations de calcul de GDH.2 [46] et il se fait en quatre étapes (Figure 13).

3.3.6.8.2.1 Création du groupe

Le premier membre à joindre le groupe en devient le créateur et le contrôleur. Le groupe se remplit au fur et à mesure que les membres s'y joignent et le dernier à s'ajouter occupe le rôle de gestionnaire du groupe.

3.3.6.8.2.2 Initialisation du groupe

Chaque membre M_i envoie sa contribution ($g^{S1***Si}$) à M_{i+1} (pour i allant de 1 à n-2). Ensuite M_{n-1} ajoute sa contribution ($g^{S1*S2***Sn-1}$) au message reçu et diffuse le nouveau message aux n-2 premiers membres. Troisièmement, chaque membre factorise sa contribution ($g^{S1*S2***Sn-1/Si}$) et l'envoie à M_n qui récupère toutes les contributions de l'étape précédente, les élève à la puissance S_n et diffuse le résultat à tous les membres du groupe pour qu'ils calculent la clé de groupe.

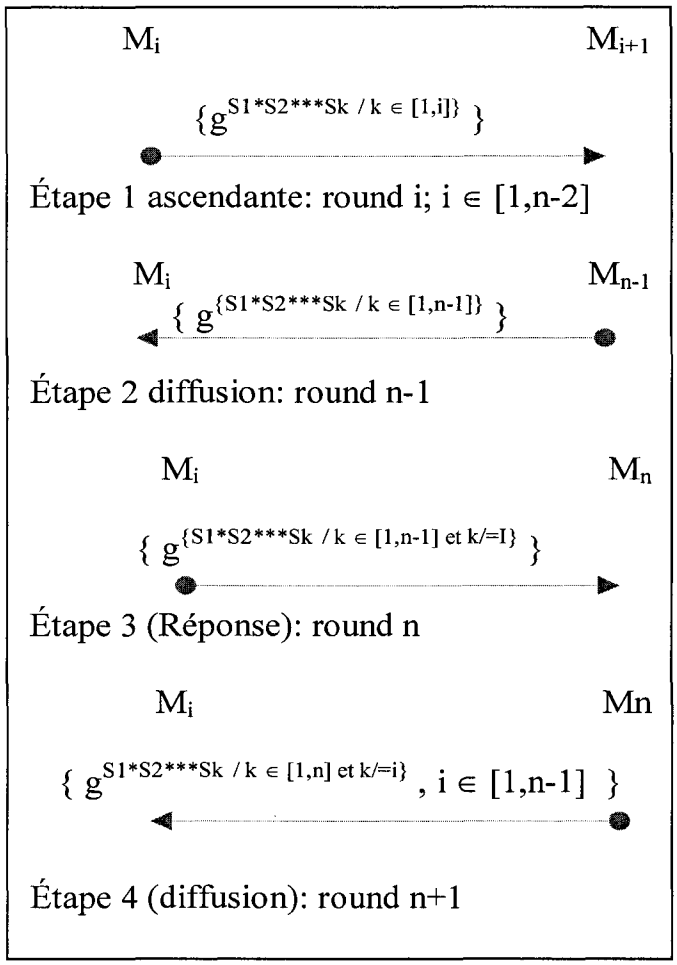

Figure 13: Le protocole GDH.3

3.3.6.8.2.3 Ajout simple

- Lors de l'ajout d'un nouveau membre M_{n+1}, le membre M_n génère une nouvelle clé individuelle et calcule de nouvelles clés intermédiaires $k_i = g^{S1*S2***Sn-1/Si}$ i ∈ [1,n-1] et les envoie à M_{n+1}.
- M_{n+1} génère sa contribution (clé), l'ajout à chaque clé intermédiaire k_i et calcule la clé $k_{n+1} = g^{S1*S2***Sn*Sn+1}$.
- M_{n+1} diffuse l'ensemble des clés $\{gS1^{*S2***Sn+1/Si} / i \in [1,n]\}$ aux autres membres du groupe.

3.3.6.8.2.4 Départ simple

- Lors du départ du participant M_p, le contrôleur M_n crée une nouvelle clé individuelle, factorise et supprime la contribution (sa clé) de M_p.
- Il calcule les différentes clés intermédiaires $\{gS1^{*S2***Sn+1/Si} / i \in [1, n]\}$ et les diffuse aux autres membres du groupe.
- Si M_n se retire du groupe, alors M_{n-1} joue le rôle de contrôleur.

Les protocoles GDH sont relativement efficaces pour les départs et le partitionnement, mais la fusion requiert un nombre de rounds égale au nombre de nouveaux membres. Ils offrent néanmoins une économie importante de la bande passante [34]. Ils permettent d'avoir un nombre minimum de messages et d'échanges comparativement à Hypercube et Octopus. Le contrôleur (chargé de la diffusion) dans GDH.2 constitue le point faible en cas d'attaque.

3.3.6.9 Approche contributive en arbre

Ces protocoles combinent les techniques de distribution en arbre présentées plus haut et les méthodes d'échange de Diffie-Hellman appliquées au groupe.

3.3.6.9.1 Le protocole TGDH

Kim, Perrig et Tsudik [37] proposent un protocole contributif de gestion de clés en arbre, le Tree-Group Diffie-Hellman (TGDH) qui vient optimiser les performances des GDH.1/2. Il est très semblable à OFT à la seule différence que chaque membre peut agir comme **sponsor** [15] dépendamment de sa position dans l'arbre. Le sponsor est chargé du calcul et de la diffusion des clés aux autres membres du groupe. Chaque membre du groupe peut calculer lui-même la clé de groupe s'il possède toutes les clés cachées dont il a besoin.

3.3.6.9.1.1 Création du groupe

Selon Baras et Striki [11] le groupe est crée au fur et à mesure de l'ajout des membres.

3.3.6.9.1.2 Ajout simple

- Détermination du point d'insertion dans l'arbre.
- Création du nœud intermédiaire et du nouveau nœud membre.
- Calcul des nouvelles clés et mises à jour du nouvel arbre.
- Diffusion par le sponsor du nouvel arbre des clés cachées.
- Mise à jour de l'arbre des clés par chaque membre et calcul de la clé de groupe.

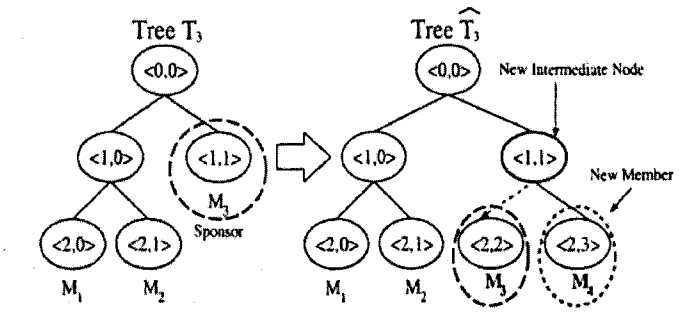

Figure 14: Mise à jour de l'arbre après ajout.

Les opérations suivantes, effectuées lors de la mise à jour de l'arbre des clés après l'ajout d'un nouveau sont illustrées à la figure 14 :

- Renommer <1,1> en <2,2>.
- Générer 2 nouveaux nœuds, l'intermédiaire <1,1> et celui du nouveau membre <2,3>.
- Faire du nœud <1,1> la clé parent de <2,2> et <2,3>.

3.3.6.9.1.3 Départ simple

Lorsqu'un participant quitte le groupe ou en est exclu, les opérations suivantes, schématisées à la figure 15, sont entreprises pour calculer une nouvelle clé de groupe :

- La suppression du nœud partant et mise à jour de l'arbre de clés.

- Le sponsor génère sa clé, calcule toutes les clés (privées et cachées) de tous ses parents jusqu'à la racine et diffuse l'arbre des clés cachées.
- Tous les membres mettent l'arbre à jour et calculent la clé de groupe.

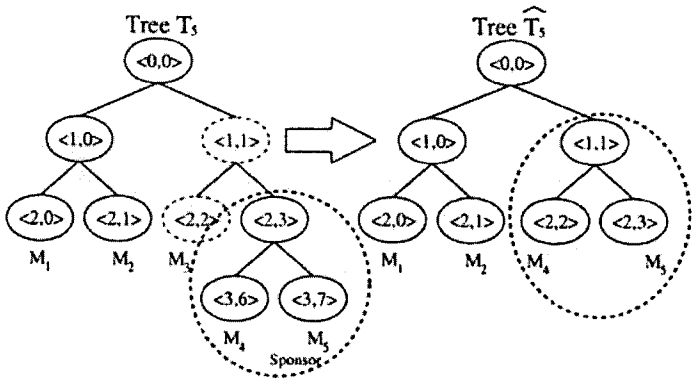

Figure 15: Mise à jour de l'arbre en cas de départ.

3.3.6.9.1.4 Séparation

La séparation des groupes est traitée comme une succession de départs simples. Tous les participants sortants sont identifiés et plusieurs sponsors sont élus comme dans le départ simple. Chaque sponsor calcule les clés dont il a la responsabilité et diffuse son arbre de clés cachées à tous les groupes. Le nouvel arbre est mis à jour par chaque participant (Figure 16).

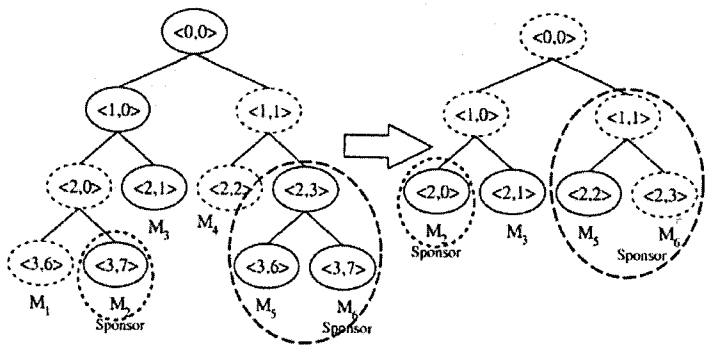

Figure 16: Arbre des clés après séparation.

3.3.6.9.1.5 Fusion

Lors de la fusion de deux ou plusieurs groupes, le sponsor (le nœud le plus à droite) de chaque groupe diffuse son arbre de clés au nouveau groupe fusionné. Chaque participant effectue la fusion des deux arbres et un nouveau sponsor calcule les clés et diffuse le nouvel arbre au groupe (Figure 17).

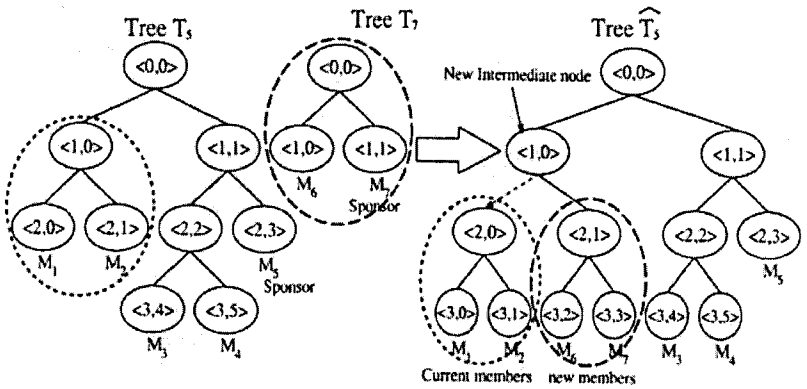

Figure 17: Arbre des clés après fusion des arbres T_5 et T_7

3.3.6.9.1.6 Choix du sponsor

Lorsque survient un événement de groupe, le sponsor est le participant le plus à droite dans le sous-arbre du nœud ayant été à l'origine dudit événement.

3.3.6.9.1.7 Structure de l'arbre des clés

L'arbre des clés est organisé de tel sorte qu'à chaque nœud de l'arbre de coordonnées <l,v>, on associe une clé privée $K_{<l,v>}$ et une cachée $BK_{<l,v>} = f(K_{<l,v>}) = g^{K_{<l,v>}} \bmod p$. Chaque participant peut calculer la clé de groupe en utilisant sa clé secrète et un ensemble de clés cachées importantes.

En fonction de ses fils de gauche et de droite, la clé cachée d'un noeud intermédiaire peut être donné par la relation :

$BK_{<l,v>} = f(K_{<l+1,2v>} * K_{<l+1,2v+1>}) = (BK_{<l+1,2v+1>})^{K_{<l+1,\,2v>}} \bmod p$.

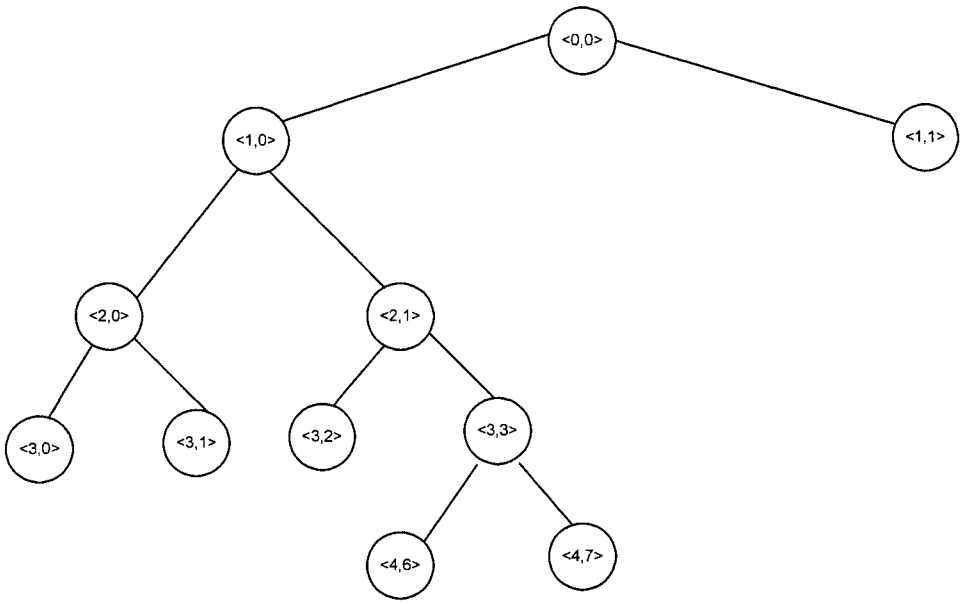

Figure 18: Structure d'un arbre des clés cachées.

$$BK_{<3,3>} = f(K_{<4,6>} * K_{<4,7>}) = (BK_{<4,7>})^{K}_{<4,6>} \bmod p$$
$$= (g^{K}_{<4,7>} \bmod p)^{K}_{<4,6>} \bmod p$$
$$= (g^{K}_{<4,7>} {}^{*K}_{<4,6>}) \bmod p$$
$$BK_{<2,1>} = f(K_{<3,2>} * K_{<3,3>}) = (g^{K}_{<3,3>} {}^{*K}_{<3,2>}) \bmod p$$
$$BK_{<1,0>} = f(K_{<2,0>} * K_{<2,1>}) = (g^{K}_{<2,1>} {}^{*K}_{<2,0>}) \bmod p$$

3.3.6.9.1.8 Avantages

- En général, le coût des ajouts et des fusions est logarithmique.
- L'initialisation du protocole n'a pas besoin de canal sécurisé entre les participants.

3.3.6.9.1.9 Inconvénients

- Lors des événements de groupe, la performance de TGDH pour le calcul de la clé de groupe dépend de la hauteur de l'arbre des clés, de son équilibre, du point d'insertion et de la situation du nœud partant.
- La séparation est une opération coûteuse.

- Un arbre profondément déséquilibré pourrait entraîner une performance de O(n).
- Dans TGDH, malgré les calculs du sponsor, les participants calculent les clés cachées en double, ce qui est coûteux en nombre d'exponentiations.
- Lors des événements en cascade, la diffusion de l'arbre des clés augmente la consommation de la bande passante.

3.3.6.10 Les approches Hybrides

De nombreux algorithmes hybrides ont été également développés et analysés. M. Striki, et S. Baras [11] proposent plusieurs modèles et les analysent. Il résulte de leurs travaux plusieurs combinaisons de méthodes existantes : distribuée-distribuée, distribuée-non contributive, distribuée-contributive, etc. Xiang-Yang Li, Yu Wang et O. Frieder [12] nous proposent une approche hybride contributive par groupage, ou tous les nœuds sont distribués dans des sous-groupes selon une répartition géographique (Figures 19 et 20). Chacun de ces sous groupe sélectionne un leader. Les différentes étapes du protocole sont les suivantes :

- La construction des sous-groupes de nœuds connectés à un leader.
- L'application d'un algorithme contributif à l'ensemble des sous-groupes (leaders+nœuds).
- Chaque leader distribue la clé de groupe à ses membres si une seule clé est requise sinon chaque sous-groupe applique son propre protocole distribué.

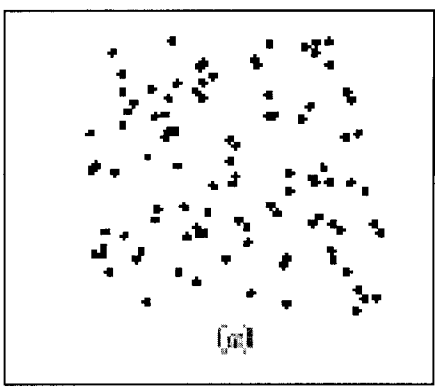

Figure 19: Sous-groupe de nœuds

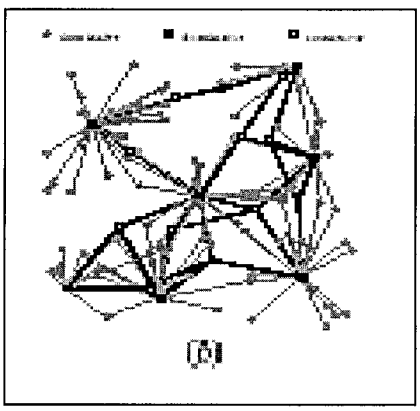

Figure 20: Ensemble de leaders

3.3.6.11 L'Approche hiérarchique à deux niveau

Dans [18], l'auteur propose l'approche hiérarchique à deux niveaux de contrôleur (Figure 21). Le niveau supérieur est occupé par le contrôleur du groupe (GSC, groupe security controler) et le deuxième niveau par les contrôleurs (leaders) de sous-groupes de membres dynamiquement élus. Ils sont gérés par le GSC qui gère à la fois les membres de tous les sous-groupes. Les sous-groupes sont indépendants de même que la gestion des clés de sous-groupes. La mise à jour est limitée au sous-groupe. L'approche hybride hiérarchique s'accommode de l'absence des ressources réseau et du changement de topologie [16].

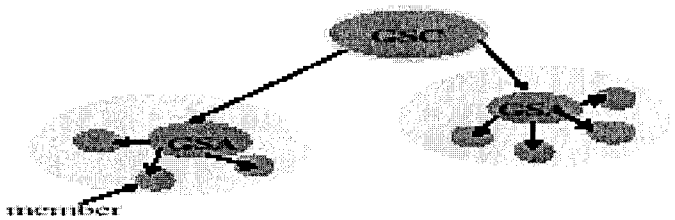

Figure 21: Approche hybride hiérarchique à deux niveaux

3.3.7 CONCLUSION

Dans ce chapitre, nous avons présenté différentes méthodes de distribution de clés de cryptage pour les communications de groupe. Nous les avons réparties dans différentes catégories : les méthodes centralisées simple et en arbre, les méthodes distribuées et celles dites contributives, linéaires ou en arbre. Nous avons dégagé les avantages et les inconvénients de chaque approche et ceux spécifiques à chaque protocole.

On a pu constater que les approches centralisées, qu'elles soient simples, en arbre ou contributives, font du contrôleur, la principale cible d'attaque. Elles sont difficiles à mettre en échelle et requièrent la disponibilité totale du serveur même lors de la séparation du réseau. Les protocoles distributifs offrent l'avantage d'être facilement

adaptables à de grands groupes. Mais la dépendance des serveurs de sous-groupe face au contrôleur central rend le groupe vulnérable si le serveur central devenait la cible des attaques. Il se pose donc le problème de la disponibilité du serveur maître. Les méthodes contributives totales permettent à tous les membres de participer également au calcul de la clé de groupe. Elles facilitent la répartition des coûts liés à la génération et au calcul des clés.

Comme nous l'avons vu dans le chapitre 2, les réseaux ad hoc sont des réseaux dynamiques point à point n'ayant aucune infrastructure préexistante (routeur, serveur de nom, entité centrale). Ils se construisent spontanément et sont la plupart du temps mobiles et temporaires. Les périphériques constituants ce type de réseau sont généralement petits et portables, donc ne disposent pas de suffisamment de mémoire, de CPU. Les connexions se font par sauts d'un périphérique à l'autre. Chaque appareil peut se déconnecter du réseau à tout moment pour des raisons de portée limitée, de déficience énergétique, à cause d'une compromission ou d'un partitionnement du réseau. Tous ces éléments posent des limitations importantes quant à la possibilité d'une diffusion globale à l'intérieur du réseau si ce n'est dans une portée restreinte. Le changement de topologie force les échanges à être rapides à cause de la volatilité des nœuds; et l'absence d'infrastructure suppose également qu'il n'y a pas de tiers partie fiable pour calculer et distribuer les clés. Par conséquent, l'utilisation d'une structure centralisée de gestion de clé de groupe est inadéquate pour un réseau ad hoc à cause de la non disponibilité éventuelle du serveur.

C'est pourquoi les méthodes contributives sont avantageuses pour les réseaux ad hoc car les parties n'ont pas besoin d'être authentifiées au préalable pour établir un lien de confiance. De plus, les méthodes contributives protègent contre les écoutes passives ou actives.

Cependant, les méthodes contributives en arbre sont plus avantageuses dans les communications de groupe, le coût des opérations de groupe est d'ordre logarithmique.

Dans le chapitre qui suit, nous présentons notre méthode de distribution de clés de cryptage pour les communications de groupe dans les réseaux mobiles ad hoc. C'est une

méthode contributive en arbre qui tente de pallier les faiblesses rencontrées dans le protocole TGDH.

Chapitre 4

La méthode GAKAP: Group Activity based Key Agreement Protocol

4.1 Introduction

Dans le chapitre précédent, nous avons fait un bref tour d'horizon des approches de distribution des clés de groupe existantes. Nous avons dégagé les avantages de l'utilisation des méthodes de distribution par accord de clés dans la génération des clés de groupe lors des communications multicast en réseau ad hoc. Les protocoles de la suite CLIQUE, GDH.2, GDH.3 et TGDH sont particulièrement intéressants. GDH.2 et GDH.3 sont avantageux quant à l'économie de la bande passante mais sont linéairement coûteux en utilisation CPU et en exponentiations en fonction du nombre d'ajouts. Quant à TGDH, il offre une performance logarithmique pour ce qui est du nombre d'exponentiations lors des ajouts et retraits simples. Mais TGDH est coûteux en bande passante et en CPU si le nombre de départs est important ou encore si les ajouts et les départs provoquent un déséquilibre de l'arbre de clés. Ce constat a amené à penser que malgré la réduction des coûts des exponentiations apportés par TGDH comparativement aux autres protocoles, ce dernier n'est pas adéquat dans les réseaux sans fil mobiles ad hoc (MANET) ou dans des réseaux très actifs. Nous pensons donc que ce protocole, s'il est utilisé lors de communication multicast dans un groupe très dynamique et dans un MANET, il pourrait se solder par un déséquilibre important dans la structure de l'arbre de clé pouvant aboutir à des exponentiation d'ordre n lors du calcul de la clé de groupe. D'autre part, si l'on est dans le même contexte, le transfert des clés à la suite des événements de groupe aboutirait à une importante consommation de la bande passante. Aussi nous pensons que la recherche d'un équilibre total de l'arbre de clés, doublé d'une réduction de la taille des messages transmis permettrait de maintenir la performance logarithmique du protocole TGDH peu importe la cascade des événements qui se produiraient au sein du groupe.

4.1.1 Définition de la méthode GAKAP

GAKAP est une méthode de distribution de clés en arbre complètement équilibré basée sur TGDH dans un réseau ad hoc pur, c'est-à-dire, un réseau ad hoc ou la mobilité et les opérations de groupe ne sont pas restreintes.

4.1.2 But de la méthode

L'objectif général de notre travail est de :

- traiter de la sécurité des communications de groupe dans un réseau sans fil mobile ad hoc (MANET),
- proposer un protocole de gestion des clés fondé sur la prise en compte de la dynamique du groupe et de la mobilité des membres,
- étendre et adapter le protocole TGDH à un environnement MANET.

Avant de détailler notre objectif, nous présentons quelques définitions qui permettent d'éclaircir notre méthode.

4.1.3 Définitions

- Activité de groupe : la notion d'activité de groupe regroupe à la fois la dynamique à l'intérieur du groupe multicast (ajout, départ...) et le degré de mobilité des membres. Dépendamment de la valeur de cette activité, on ajuste la structure de l'arbre des clés du groupe multicast.
- Seuil d'activité du groupe: il s'agit d'une valeur probabiliste limite définissant la valeur au-delà de laquelle la dynamique du groupe aboutirait à un déséquilibre de l'arbre des clés, entraînant des exponentiations d'ordre n et une consommation importante de la bande passante.

- Initiateur du groupe : il s'agit du membre qui initie la session multicast. Il construit la liste des participants ou liste de contrôle d'accès (ACL), l'arbre des clés cachées et les diffuse aux autres participants.
- Contrôleur initial: il s'agit du membre chargé de coordonner (assurer) l'établissement de la clé de groupe initiale. Cela peut être l'initiateur du groupe ou un autre membre dédié.
- Contrôleur temporaire: lors de certains événements de groupe, pour uniformiser la structure de l'arbre des clés, ce contrôleur a la responsabilité de construire l'arbre et de le diffuser aux autres participants.
- Sponsor: c'est tout participant qui joue le rôle de contrôleur à la suite d'un événement de groupe. Il est chargé de la distribution des clés en vue du renouvellement de l'arbre.
- Mise à jour des clés : à la suite d'une opération de groupe (ajout, départ), certaines clés sont compromises et doivent être remplacées afin d'assurer la confidentialité des messages (antérieurs ou à venir), le changement de la valeur de ces clés est leur mise à jour.
- Échange DH : ensemble de deux messages échangés simultanément par deux participants.
- Round synchrone : chaque participant peut arbitrairement envoyer plusieurs paquets pendant une unité de temps et recevoir arbitrairement plusieurs paquets au début d'un round [29][20].
- Round simple : chaque participant peut envoyer ou recevoir tout au plus un message pendant chaque round [29][20].
- Participant : désigne tout membre du groupe prenant part à une session multicast.

L'objectif spécifique de notre méthode est de trouver la valeur idéale α de l'activité de groupe pour laquelle la distribution des clés lors d'une communication multicast en réseau Manet serait optimale et permettrait sinon de diminuer, du moins, de maintenir le nombre d'exponentiations logarithmiques nécessaires lors de l'établissement de la clé de

groupe malgré les influences conjuguées de la mobilité des membres et de la dynamique des opérations de groupe.

4.1.4 Notations

Symboles	Description
$\{m\}_k$	Le message m est encrypté avec la clé k
$\{M_i\}$	Participant M d'indice i.
BK_i	Clé cachée du participant d'indice i.
$\{BK^*_{Si}\}$	Ensemble de clés cachées envoyées par le sponsor s_i.
$\{M_1..M_n\}$	Groupe de participants.
GK	Clé de groupe
*	Multiplication entre un scalaire et un point
.	Multiplication entre deux scalaires

4.2 Types de paquets

Nous décrivons dans les pages suivantes, les différents types de paquets utilisés lors des l'échanges des clés de communications entres les participants d'une session multicast. Les paquets sont envoyés sous forme de message crypté contenu dans un paquet général.

4.2.1 Description des paquets

Paquet : GAKAP

Type_proto	Taille_totale
SOURCE	
DESTINATION	
DONNÉES	

Fonction : c'est le paquet général du protocole GAKAP, celui dans lequel sont encapsulés tous les autres types de paquets. Ses champs contiennent l'adresse de la source, l'adresse de la destination et le type de paquet contenu dans le paquet général.

Utilisé : À chaque envoie de paquet.

Type de message : Unicast ou Multicast.

Champs

Type_proto : entier (chaîne de bits) indiquant le type de protocole ou de paquet contenu dans le champ données.

Taille_totale : désigne la taille totale du paquet.

SOURCE : adresse ou identificateur (nom) de la source du paquet.

DESTINATION : adresse(s) ou identificateur(s) de la destination du paquet.

DONNÉES : représente un type de paquets ci-dessous.

Paquet : LEAVE

Type_mes	Taille
DONNÉES	

Fonction : désigne le type de paquet de départ qui est envoyé.

Utilisé : lors des requêtes de départ et lors des mises à jour à la suite du départ d'un membre.

Type de message : Multicast.

Champs

Type_mes : entier indiquant le type de message. Il peut s'agir d'une requête de départ (**leave_req**) ou d'une requête de mise à jour à la suite d'un départ simple (**leave_update**).

Taille : donne la taille du paquet de départ.

DONNÉES : Il peut s'agir d'un des paquets BKEY ou BKEYS décrits plus loin dans ce paragraphe.

Paquet : ADD

Type_mesg	Taille
DONNÉES	

Fonction : indique aux participants que le paquet contient un message d'ajout simple.

Utilisé : lors des requêtes d'ajouts simples.

Type de message : Unicast ou Multicast.

Champs :

Type_mesg : entier indiquant le type de message. Il peut s'agir d'une requête d'ajout (**add_req**) ou d'une requête de mise à jour à la suite d'un ajout simple (**add_update**) ou d'un message contenant l'arbre des clés envoyé au nouveau membre.

Taille : donne la taille du paquet d'ajout.

DONNÉES : il peut s'agir d'un des paquets BKEY, BKEYS ou TREE décrits plus loin.

Paquet : ADD_G

Type_mesg	Taille
DONNÉES	

Fonction : Indique aux participants que le paquet contient un message d'ajout multiple.

Utilisé : lors des requêtes d'ajout multiple et lors des différentes mises à jour à la suite d'un ajout multiple.

Type de message : Multicast.

Champs :

Type_mesg : entier indiquant le type de message. Il peut s'agir d'une requête d'ajout multiple (**add_g_req**) ou d'une requête de mise à jour ou d'un remplacement de l'arbre de clé envoyé par le contrôleur temporaire (**add_g_update_tree**) ou d'un message de mise à jour des clés (**add_g_update**) envoyé par les autres sponsors à la suite d'un ajout multiple.

Taille : donne la taille du paquet d'ajout.

DONNÉES : Il peut s'agir d'un des paquets BKEYS ou TREE décrits plus loin.

Paquet : LEAVE_G

Type_mesg	taille
DONNEES	

Fonction : indique aux participants que le paquet contient un message de départ multiple.

Utilisé : lors des requêtes de départ multiple et lors des différentes mises à jour à la suite d'un départ multiple.

Type de message : Multicast.

Champs :

Type_mesg : entier indiquant le type de message. Il peut s'agir d'une requête de départ (**leave_g_req**) envoyé par le contrôleur temporaire ou d'un message de mise à jour des clés (**leave_g_update**) envoyé par les autres sponsors à la suite du départ multiple.

Taille : donne la taille du paquet d'ajout.

DONNÉES : Il peut s'agir d'une liste des participants sortants ou d'un paquet BKEYS décrits plus loin dans cette partie.

Paquet: BKEY

Key_ID	Taille	Key_value

Fonction : désigne un paquet contenant une clé cachée.

Utilisé : lors des requêtes des différents événements de groupe précédemment citées à l'intérieur du champ de données.

Type de message : Unicast ou Multicast.

Champs :

Key_ID: chaîne de caractères (ou entier) représentant le nom (ou l'identifiant) de la clé.

Taille : donne la taille du paquet BKEY.

Key_value: entier indiquant la valeur de la clé.

Paquet: BKEYS

Key_ID	Taille	Key_value
Key_ID	Taille	Key_value
Key_ID	Taille	Key_value
...	...	...

Fonction : Tableau contenant l'ensemble des clés cachées devant être modifiées ou ayant été modifiées par le sponsor.

Utilisé : lors des requêtes d'ajout, de départ multiple, de fusion ou de séparation et contenu à l'intérieur du champ de données.

Champs :

Key_ID: chaîne de caractères (ou entier) représentant le nom (ou l'identifiant) de la clé.

Taille : donne la taille du paquet BKEY.

Key_valeur: entier indiquant la valeur de la clé.

Paquet : TREE

Nombre de niveau	Nombre de noeuds
BKEYS	

Fonction: contient l'arbre des clés.

Utilisé: lors des requêtes d'ajout, de fusion ou de séparation.

Type de message : Multicast.

Champs :

Nombre de niveaux: entier désignant la profondeur de l'arbre des clés.

Nombre de noeuds: entier désignant la taille totale de l'arbre des clés et permettant la construction de ce dernier.

Paquet : PARTITION

sponsor_ID	type_mesg
TREE	

Fonction : ce paquet indique aux participants la séparation du groupe en deux ou plusieurs groupes.

Utilisé : lors de la séparation du groupe multicast.

Type de message : Multicast.

Champs :

Sponsor ID : identificateur (nom) du participant demandant la séparation.

Type_mesg : il s'agit du type de message de partitionnement, une requête **part_req.**

TREE : paquet TREE contenant le sous arbre des participants voulant se séparer.

Paquet : MERGE

Sponsor_ID	type_ mesg
TREE	

Fonction : ce paquet indique aux participants la fusion de deux ou plusieurs groupes.

Utilisé : lors du partitionnement du groupe multicast.

Type de message : Multicast.

Champs :

Sponsor ID : identificateur (nom) du participant demandant la fusion de son groupe.

Type_mesg : il s'agit du type de message de fusion, une requête **merge_req.**

TREE : paquet TREE contenant le sous arbre des participants voulant se fusionner.

Paquet : REFRESH

Sponsor_ID	No_seq
BKEYS	

Fonction : paquet désignant les nouvelles clés suite au rafraîchissement de la clé de groupe.

Utilisé : Ce paquet est utilisé pour indiquer le rafraîchissement de la clé de groupe à la suite du dépassement de la durée limite d'utilisation de la clé. Le paquet est envoyé s'il n'y a pas eu d'événement de groupe.

Champs

Sponsor_ID : c'est l'identificateur du sponsor ayant procédé au rafraîchissement de la clé de groupe, c'est le dernier sponsor élu lors du dernier événement survenu dans le groupe.

No_seq : c'est le numéro de séquence du rafraîchissement.

Bkeys : champ contenant les clés cachées ayant été rafraîchies.

4.2.2 Le stockage

Chaque participant stocke l'arbre des clés cachées ainsi que les clés secrètes de tous les nœuds intermédiaires parents sur son chemin jusqu'à la racine.

4.2.3 Clé glissée

Une clé est glissée lorsqu'à la suite du départ d'un des jumeaux d'une paire de participants, la nouvelle clé du participant restant devient celle du nœud parent.

4.2.4 Clé cachée

La clé cachée d'un membre est une clé publique obtenue par application d'une fonction à sens unique sur sa clé privée.

4.2.5 La constante d'activité

Le calcule de la constante d'activité se fait en déterminant le rapport de la somme du nombre d'ajouts simples, multiples et de reconnexions survenus dans le groupe pendant une période de temps déterminée sur la somme du nombre de départs simples, multiples

ou de déconnexions survenus pendant la même période. La constante d'activité est donnée par la relation :

$$\alpha = \frac{\sum(Ajouts + \text{Re}connexions)}{\sum(Depart + Déconnexions)}$$

4.3 La cryptographie par courbes elliptiques

Depuis quelques années, les courbes elliptiques sont de plus en plus utilisées comme méthode cryptographique. Elles ont été utilisées entre autres dans les cartes à puces. Elles suscitent un intérêt particulier dans le monde des périphériques sans fil comme les PDA.

4.3.1 Définition

La cryptographie par courbes elliptiques opère sur les points d'une courbe elliptique. L'opération de base est la multiplication de points d'une courbe avec de grands nombres entiers : k*P. Elle revient en effet à une addition du point P, k fois. La cryptographie par courbe elliptique pose elle aussi l'épineux problème de la résolution du logarithme discret qui consiste à retrouver k connaissant P=k*Q. La sécurité de cette méthode cryptographique réside dans le choix de la courbe elliptique et de la difficulté que celle-ci pose à résoudre le problème du logarithme discret. Il existe quelques algorithmes qui permettent d'effectuer ce choix.

4.3.2 Courbes elliptiques sur le corps fini GF(p)

Soit $p > 3$ un nombre premier et a, b $\in$ GF(p) tel que $4a^3 + 27b^2 \neq 0$. Une courbe elliptique E sur GF(p) est définie par les paramètres a et b, l'ensemble des couples (x, y) $\in$ GF(p) x GF(p) satisfaisant l'équation $y^2 = x^3 + ax + b$ et un point à l'infini O.

4.3.2.1 Les propriétés des courbes elliptiques GF(p)

Corollaire : l'ensemble des points P $\in$ E forme un groupe respectant les propriétés additives suivantes :

$O + O = O$

$(x, y) + O = (x, y)$

$(x, y) + (x,-y) = O$

4.3.2.1.1 Additionner deux points $P_1(x_1, y_1)$ et $P_2(x_2, y_2)$ tel que $x_1 \neq x_2$

$(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$

$\lambda = y_2 - y_1 / x_2 - x_1$

$x_3 = \lambda^2 - x_1 - x_2$

$y_3 = \lambda(x_1 - x_3) - y_1$

4.3.2.1.2 Doubler un point $P(x_1,y_1)$

$(x_1, y_1) + (x_1, y_1) = (x_3, y_3)$

$\lambda = 3x_1^2 + a / 2y_1$

$x_3 = \lambda^2 - 2x_1$

$y_3 = \lambda(x_1 - x_3) - y_1$

4.3.3 Courbes elliptiques sur le corps $GF(2^k)$

Une courbe elliptique non super singulière E sur un corps fini $GF(2^k)$ est définie par les paramètres a et b $\neq$ 0, l'ensemble des couples (x,y) $\in$ $GF(2^k)$ x $GF(2^k)$ satisfaisant l'équation

$y^2 + xy = x^3 + ax^2 + b$ et un point à l'infini O.

4.3.3.1 Les propriétés des courbes elliptiques GF(2^k)

$O + O = O$

$(x, y) + O = (x, y)$

$(x, y) + (x, x+y) = O$

4.3.3.1.1 Additionner deux points $P(x_1 , y_1)$ et $Q(x_2 , y_2)$ tel que $x_1 \neq x_2$

La somme de deux points P et Q d'une courbe elliptique (Figure 22) est régie par les propriétés suivantes :

$(x_1,y_1) + (x_2, y_2) = (x_3, y_3)$

$\lambda = y_1 + y_2/ x_1 + x_2$

$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a$

$y_3 = \lambda(x_1 + x_3) + x_3 + y_1$

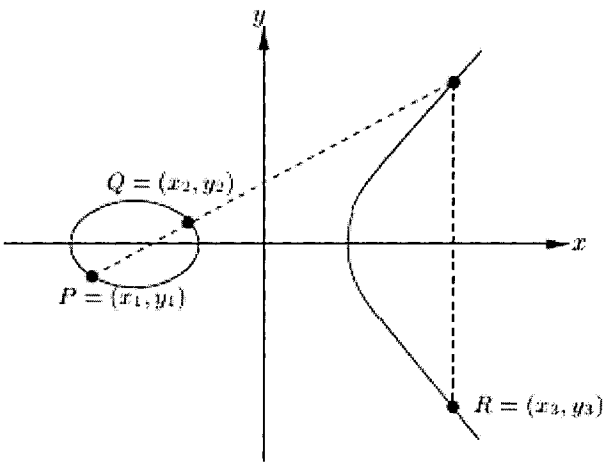

Figure 22: Addition de deux points d'une courbe elliptique

4.3.3.1.2 Doubler un point $P(x_1, y_1)$ de la courbe elliptique

Le doublage d'un point de la courbe elliptique (Figure 23) est donné par les propriétés ci-dessous :

$(x_1,y_1) + (x_1,y_1) = (x_3,y_3)$

$\lambda = x_1 + y_1/x_1$

$x_3 = \lambda^2 + \lambda + a$

$y_3 = x_1^2 + (\lambda + 1)x_3$

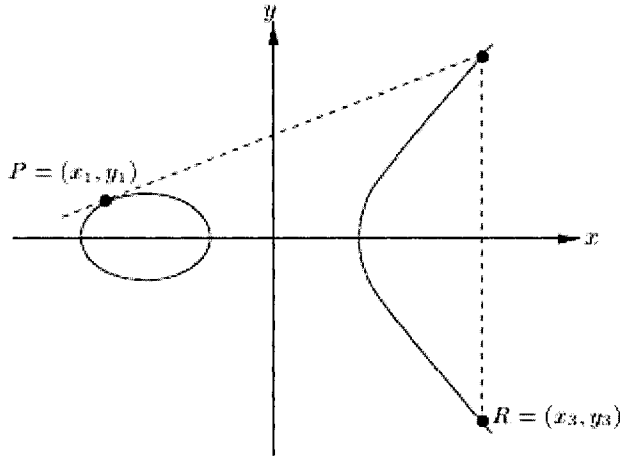

Figure 23: Doublage de point d'une courbe elliptique

4.4 Méthode de calcul de la chaîne d'addition

Pour calculer e * Q, il faut déterminer le nombre d'additions possibles pour un entier k donné. Cela veut dire trouver la **chaîne d'additions** correspondante. La chaîne d'additions peut être obtenue de plusieurs façons : par la méthode binaire ou la méthode n-aire.

Une chaîne additive est une séquence d'entiers a_0 a_1 a_2 … a_r avec $a_0 = 1$ et $a_r = k$ tel que tout $a_k = a_i + a_j$ pour $0 < i, j < k$.

4.4.1 Méthode Binaire

Pour calculer P = s*Q

- Faire la décomposition binaire de s
- Appliquer l'algorithme suivant :

  Si $s_{k-1} = 1$ alors Q := P

  Sinon Q := O

  Pour i allant de k-2 à 0 faire

Q :=Q+Q

Si $s_i = 1$ alors Q := Q+P

Retourner Q

4.4.2 Application des courbes elliptiques à la méthode Diffie-Hellman

Nous avons vu au chapitre 3 l'échange de clés par la méthode de Diffie-Hellman. La méthode classique de Diffie-Hellman est basée sur un groupe multiplicatif modulo p. Celle utilisée avec les courbes elliptique appelée ECDH (Figure 24) est basé sur un groupe additif des points d'une courbe elliptique et opère comme suit :

1. Permet d'établir une clé partagée entre 2 parties
2. Considère que les paramètres a, b et le point générateur P sont préalablement définis.
3. A et B s'entendent sur une courbe elliptique E et un point P
4. A calcule Q = k*P
5. B calcule Q' = k'*P
6. A et B s'échangent Q et Q'
7. Ils calculent la valeur commune Q''=k.k'*P

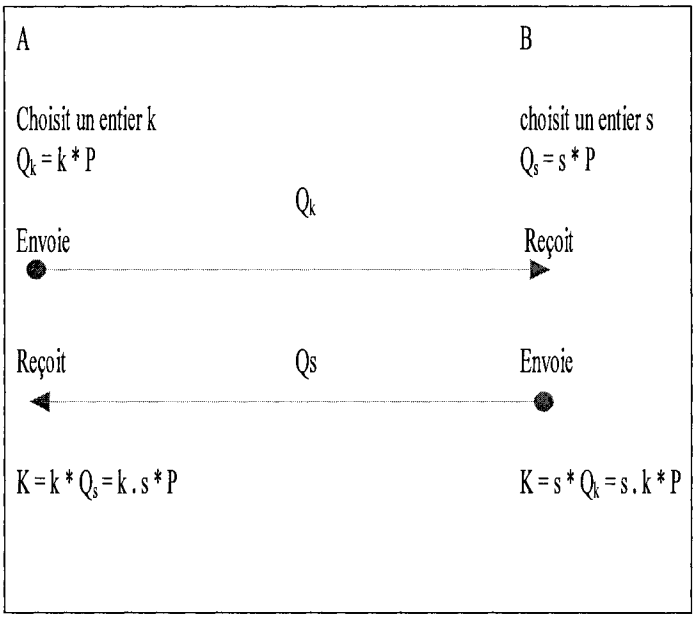

Figure 24: Échange DH des points d'une courbe elliptique.

4.4.3 Génération de la clé DSA par courbes elliptiques

L'obtention de la clé DSA par application des courbes elliptiques (ECDSA) et permettant la signature numérique des messages s'effectue de la façon suivante :

- choisir un entier d∈ [2, n-2] ;
- calculer Q = d*P ;
- les clés privées et publiques sont d et (E, P, n, Q).

4.4.4 Génération de la signature ECDSA

La signature numérique d'un message à l'aide des courbes elliptiques suit les étapes suivantes:

1. choisir un nombre entier k∈ [2, n-2] ;
2. calculer k*P=(x1, y1) et r = x1 mod n .
3. si x1∈GF(2^k) alors x1 est une décomposition binaire.

4. si r =0 alors aller à l'étape 1.
5. calculer k-1 mod n.
6. calculer s = k-1(H(m)+dr)mod n ou H=SHA-1.
7. si s =0, aller à 1.
8. la signature du message m est (r, s).

4.4.5 Structure de l'arbre des clés

L'arbre des clés est organisé de tel sorte qu'à chaque nœud de l'arbre de coordonnées <l,v>, on associe une clé privée $K_{<l,v>}$ et une **CLÉ** cachée $BK_{<l,v>} = f(K_{<l,v>}) = K_{<l,v>} * P$. Chaque participant peut calculer la clé de groupe en utilisant sa clé secrète et un ensemble de clés cachées importantes.

En fonction de ses fils de gauche et de droite, la clé cachée d'un noeud intermédiaire peut être donné par la relation :

$BK_{<l,v>} = f(K_{<l+1,2v>} * K_{<l+1,2v+1>}) = (BK_{<l+1,2v+1>} . K_{<l+1, 2v>}) * P.$

4.5 Choix de la courbe elliptique

Dans notre protocole, pour réduire au maximum la taille des clés de cryptage, le temps de calcul relié à la génération et à l'échange des clés de cryptage, nous avons choisi d'utiliser la méthode de chiffrement par courbes elliptiques appliquée à l'échange de Diffie-Hellman. Le type de courbes choisies sont celles définies sur le groupe $GF(2^k)$ car l'espace et le temps des opérations arithmétiques sur ce groupe sont plus efficaces.

4.6 Avantages et inconvénients des courbes elliptiques

4.6.1 Avantages

- Posent le problème du logarithme discret sur les courbes elliptiques.
- Ces dernières sont définies sur des groupes peu connus comparativement à (Z/pZ).

- Offre une taille réduite de la clé pour un même degré de sécurité que RSA (164 bits %1024).
- Utilise des opérations mathématiques simples dans le calcul des clés (+,-, x, /).
- La taille des groupes de points est limitée mais suffisamment élevée.

4.6.2 Inconvénients

- La théorie des courbes elliptiques est complexe et récente.

4.7 Choix du sponsor

À la suite d'une opération de groupe, l'un des membres s'investit du rôle de contrôleur de groupe (le serveur de clés) ou sponsor. Il est choisi de la façon suivante (Figure 25) :

1. Le nœud voisin du nœud sortant ou entrant, s'il existe, devient le sponsor.
2. Le nœud ayant le plus proche lien de parenté avec le nœud entrant ou sortant devient le sponsor.

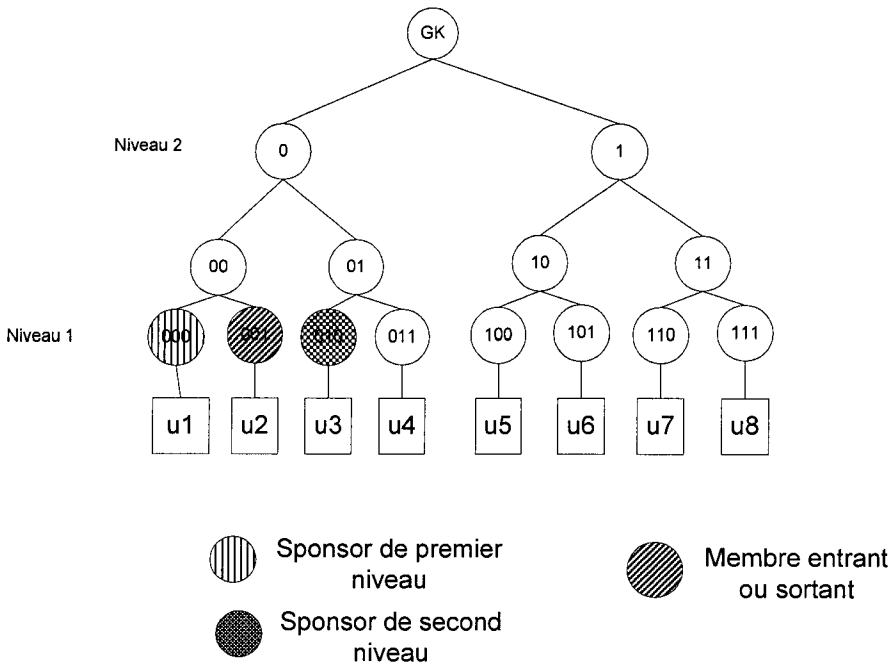

Figure 25 : Étapes du choix du sponsor.

4.8 Les opérations sur le groupe

Du début de la session de groupe jusqu'à la fin, le groupe multicast subit de nombreux changements. Les événements qui sont à l'origine de ces changements constituent ce que l'on appel les opérations de groupe. Les lignes qui suivent présentent les différentes opérations du groupe multicast et la façon dont la gestion des clés de cryptage est traitée.

4.8.1 Création du groupe

- L'initiateur dresse la liste des participants, construit l'arbre des clés cachées et les diffuse aux membres du groupe.
- Pour chaque round i, i allant de 1 à h (avec $h=\log_2 N$)
    - $N/2^i$ des membres du groupe deviennent des sponsors.
    - Chaque sponsor calcule le maximum de clés possibles sur son chemin et les diffuse.
    - Il remplace la clé manquante de niveau l par celle de son fils de niveau l+1 (figure 27)
- Fin pour
- Au round h+1, chaque membre calcule la clé de groupe
- Chaque membre calcule la constante d'activité initiale du groupe en fonction de la mobilité des nœuds et du nombre des événements de groupe qui sont survenus pendant une période donnée.

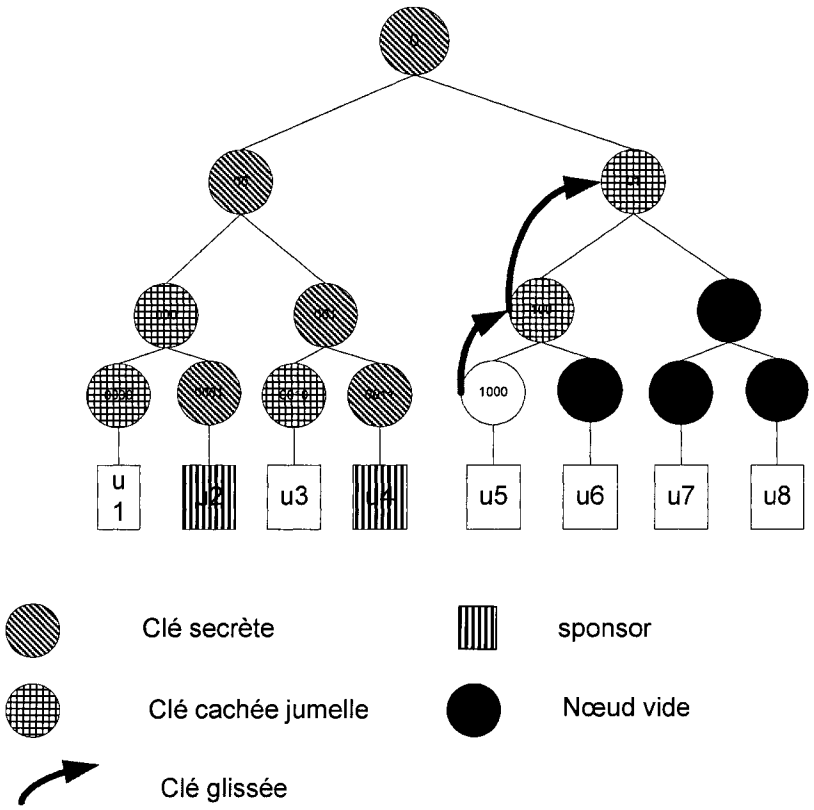

Figure 26: Initialisation du protocole GAKAP

Description de l'initialisation du protocole GAKAP présentée à la figure 26 :

- U2 et U4 sont des sponsors.
- Les nœuds [0011], [001],[00] forment le chemin du sponsor U_4.
- Les nœuds [0010], [000] et [01] forment le chemin complémentaire au chemin du sponsor.

Pour calculer la valeur du nœud [001], U_4 se sert de la clé secrète du nœud [0011] et de la clé cachée du nœud [0010]. Pour calculer celle du nœud [00], U_4 se sert de la clé secrète du nœud [001] et de la clé cachée du nœud [000].

4.8.2 Ajout simple

Lors de l'ajout simple, un membre, à la fois, envoie une requête multicast d'ajout comprenant sa clé cachée.

4.8.2.1 Opérations chez le sponsor

- Il change sa clé secrète et met son arbre de clés à jour en calculant toutes les clés et clés cachées de ses parents sur son chemin jusqu'à la racine (Figure 27);
- Il diffuse un paquet ADD à tous les autres membres. Ce paquet contient les clés cachées qui ont été modifiées. Il envoie un paquet TREE au nouveau membre, la liste des participants et la constante d'activité du moment.

4.8.2.2 Opérations chez les autres membres

- Chaque membre détermine le point d'insertion, c'est-à-dire, le premier nœud libre en parcourant les feuilles de l'arbre des clés de gauche à droite.
- Si l'arbre des clés est déjà complet ou s'il se remplit à la suite de l'ajout et si la constante d'activité du groupe est inférieur ou supérieur au seuil, chaque membre du groupe construit un nouvel arbre ayant pour sous arbre gauche l'arbre de clés précédent et un nouveau sous arbre composé de nœuds vides.

4.8.2.3 Opérations chez le nouveau membre

- Il calcule les clés manquantes de tous les parents sur son chemin jusqu'à la racine.

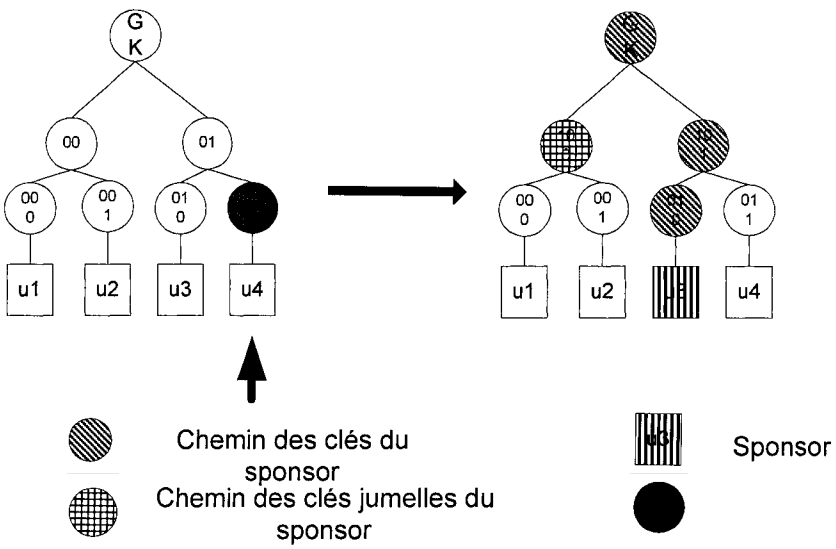

Figure 27 : Exemple d'arbre à la suite d'un ajout simple

4.8.2.4 Exemple de détail des opérations à la suite d'un ajout simple

- Diffusion d'un paquet ADD (add_req, bkey).
- Suppression par chaque participant des clés en {[0], [01], [010]}.
- Génération de la nouvelle clé [010] par le sponsor.
- Le sponsor calcule ses clés secrètes et clés cachées (bkeys) {[0], [01], [010]}.
- Il diffuse ensuite un paquet UPDATE_KEY ({[0], [01], [010]}).
- Chaque participant remplacement les bkeys {[0], [01], [010]} ainsi obtenues.
- Le sponsor envoie un paquet TREE de l'arbre des bkeys au nouveau membre.

4.8.3 Départ simple

Lors du départ simple, un seul participant quitte le groupe à la fois en envoyant une requête de départ comprenant son nom (ID).

4.8.3.1 Opérations du sponsor

- Il génère une nouvelle contribution et met son arbre de clés à jour (figure 28).
- Il diffuse ensuite à tous les autres membres un paquet LEAVE contenant les clés cachées modifiées.
- Pendant une période de temps déterminée, si l'un des sous arbres est vide et si la constante d'activité est différente du seuil, ce sous arbre est détruit et le sous arbre restant devient le nouvel arbre des clés.

4.8.3.2 Opérations au niveau des autres membres

- Chaque participant détermine le nœud sortant et le **libère**.
- Il remplace les clés à changer par celles contenues dans le paquet LEAVE reçues du sponsor.
- Pendant une période de temps déterminée, si l'un des sous arbres est vide et si la constante d'activité est différente du seuil, ce sous arbre est détruit et le sous arbre restant devient le nouvel arbre des clés.

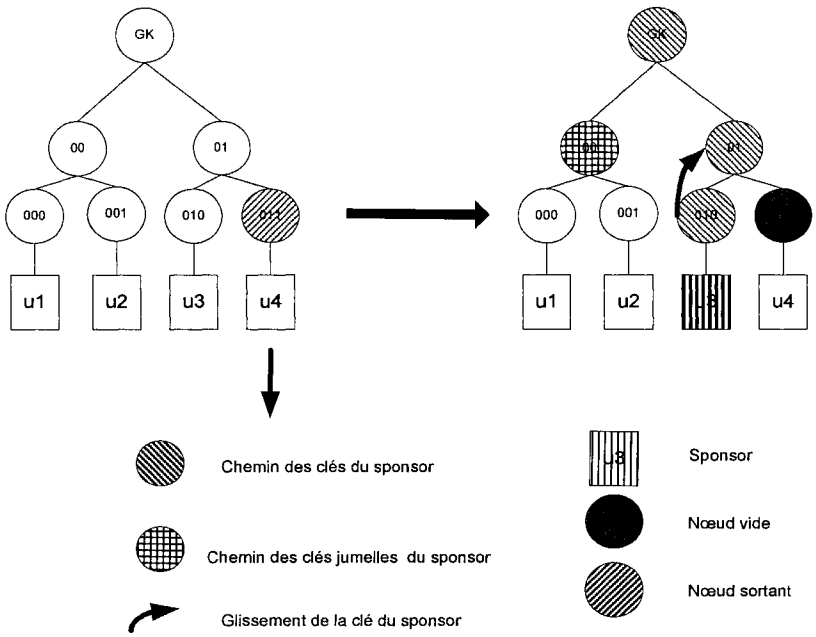

Figure 28 : exemple d'arbre de clés lors du départ simple

4.8.3.3 Exemple de détail des opérations à la suite du départ simple

- Le participant sortant diffuse un paquet LEAVE (leave_req, [011]).
- Chaque membre supprime la clé [011] du sortant.
- Tous les membres suppriment des clés cachées en {[01], [010]}.
- Le sponsor génère une nouvelle clé [010].
- Il calcule les clés secrètes et les clés cachées (bkeys) {[010]}.
- Comme la clé cachée en [011] est absente, il glisse celle en [010] en [01] et utilise la [00] pour calculer la clé de groupe [GK].
- Le sponsor diffuse un paquet UPDATE_KEY {[010], [01]}.
- Tous les membres remplacent les clés cachées {[010], [01]}.

4.8.4 Ajouts multiples

L'opération est presque identique à un ajout simple, à la seule différence qu'elle implique l'ajout de plusieurs membres et se passe comme suit. Lors de l'ajout multiple, plusieurs membres demandent à être ajoutés au groupe de façon quasi synchrone dans

un intervalle de temps prédéterminé. Les nouveaux participants (demandeurs) connaissent seulement leur contribution (clé secrète) et leur clé cachée. Un des participant déjà inscrit sera élu contrôleur temporaire.

4.8.4.1 Opération chez le contrôleur temporaire

Lors de l'ajout multiple, le sponsor le plus à droite devient le contrôleur temporaire et effectue les opérations suivantes :

- Il vérifie les différents points d'insertion disponibles en parcourant l'arbre des clés de gauche à droite. Si le nombre de nœuds disponibles est suffisant, il insère les nouveaux participants; sinon, un nouvel arbre est construit de la même manière que dans le cas de l'ajout simple.
- Il choisit une nouvelle contribution et calcule ses clés secrètes et cachées.
- Il fabrique un paquet TREE contenant l'arbre des clés cachées et le diffuse à tous les participants y compris les nouveaux. Il envoie aussi la constante d'activité du moment.

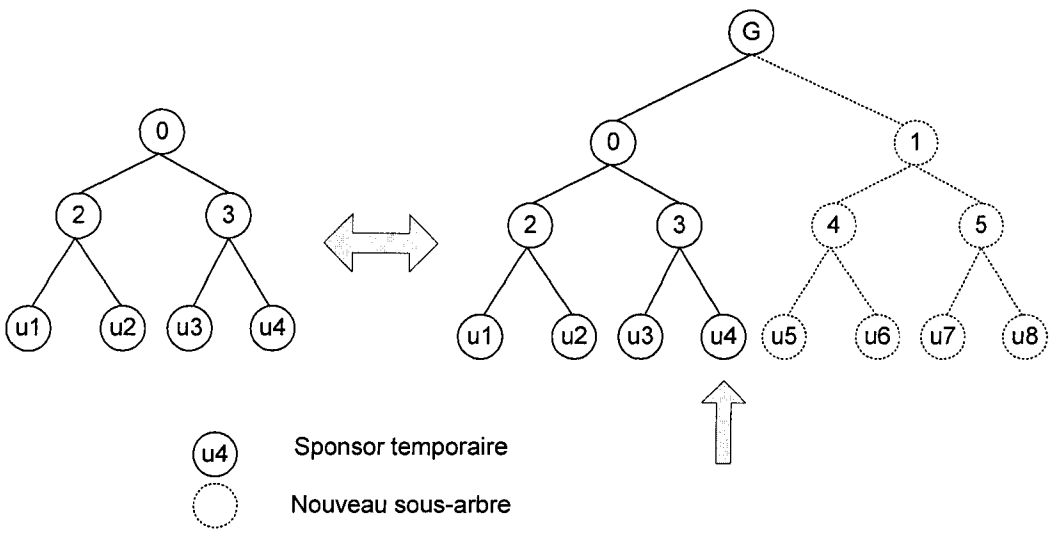

Figure 29: Construction et déconstruction d'un sous-arbre lors de l'ajout ou du départ d'un membre.

4.8.4.2 Opérations chez les autres sponsors

Pour chaque paire de nouveaux participants, l'un d'eux est appelé à jouer le rôle de sponsor.

Chaque sponsor met son arbre de clés à jour en calculant toutes les clés secrètes et clés cachées de ses parents sur son chemin jusqu'à la racine et il diffuse ensuite les clés cachées modifiées à tous les autres membres. Cette opération se fait en plusieurs étapes suivant un intervalle de temps déterminé (Figure 30).

4.8.4.3 Opérations chez les autres participants

- Ils déterminent les clés qui doivent être mises à jour.
- Ils les remplacent par celles reçues des sponsors.
- Chaque participant calcule la clé de groupe.

À la fin de l'ajout multiple, tous les membres du groupe possèdent le même arbre de clés, la même clé de groupe et la même liste de participants. Le rôle du contrôleur temporaire est aboli jusqu'au prochain ajout multiple.

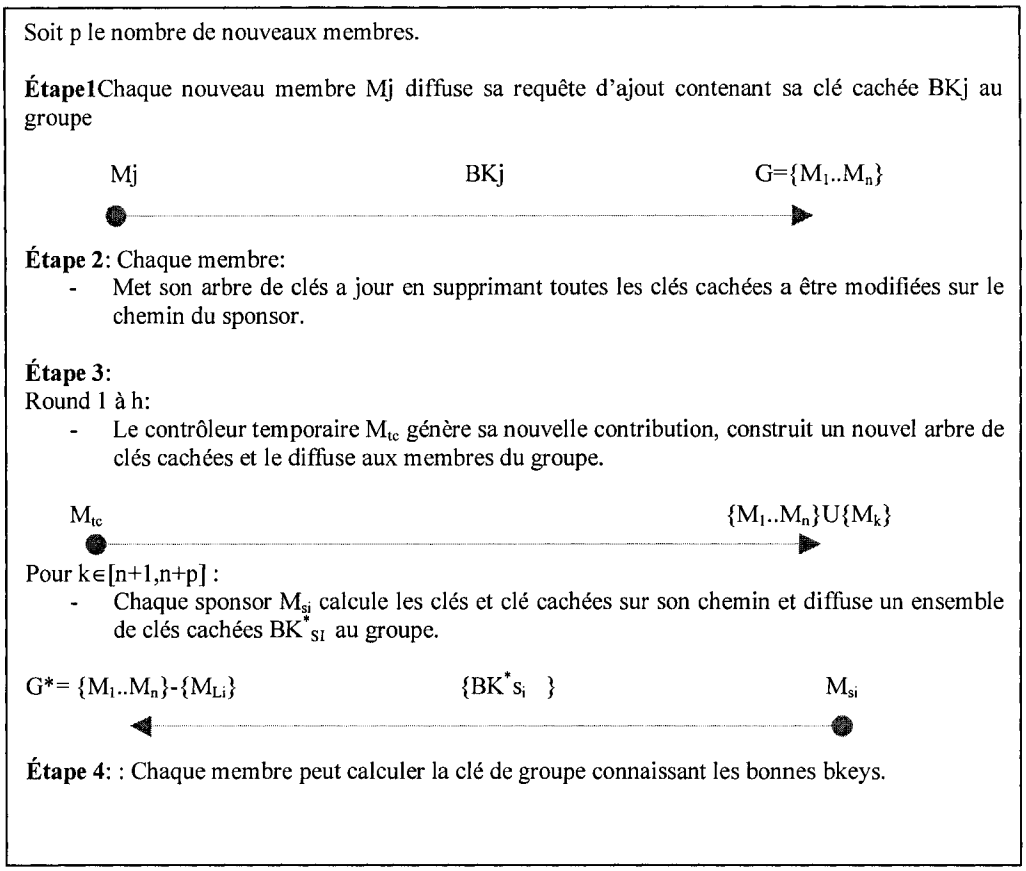

Figure 30: Interaction lors du protocole d'ajout multiple.

4.8.5 Départs multiples

Lors du départ multiple, plusieurs demandent de retrait sont faites en même temps par différents membres du groupe, il s'ensuit différentes opérations décrites à la figure 31.

4.8.5.1 Opérations chez le sponsor

- Il identifie tous les participants sortants.
- Il génère une nouvelle clé et calcule, tant qu'il peut, les clés secrètes et cachées de tous les nœuds parents sur son chemin jusqu'à la racine.
- Il encapsule les clés cachées du paquet BKEYS dans un paquet général GAKAP et le diffuse aux autres membres du groupe.

- Il remplace toutes les autres clés cachées ayant été modifiées contenues dans les paquets BKEYS qu'il reçoit des autres sponsors.
- Il calcule la clé de groupe.

4.8.5.2 Opérations chez les autres participants

- Chaque participant libère les nœuds des participants sortants de même que les nœuds parents intermédiaires qui doivent l'être.
- Il met l'arbre à jour en remplaçant toutes les clés cachées ayant été modifiées par les sponsors et contenus dans les paquets BKEYS.
- Ils calculent la clé de groupe.
- À la suite des départs, si l'un des sous arbres droits ou gauches est vide et si la constante d'activité le permet, le sous-arbre non vide restant devient le nouvel arbre des clés.

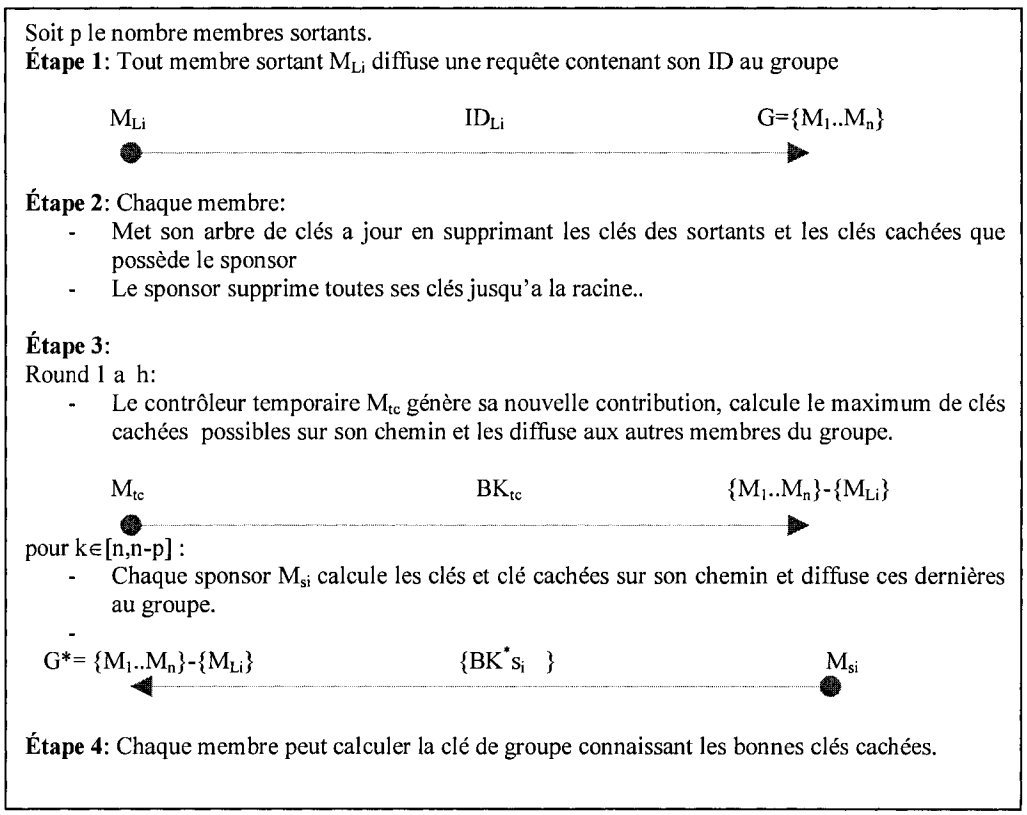

Figure 31: Interaction lors du départ multiple

4.8.6 Départs accidentels

Si un ou plusieurs participants quittent accidentellement le groupe à la suite d'une déconnexion due à la mobilité (i.e. changement de topologie), alors il faut trouver un mécanisme pour vérifier la présence des membres. Cette vérification va se faire au niveau des nœuds participants à intervalle régulier. Le protocole se comporte comme dans les cas de départ simple ou multiple. Les **départs accidentels** pourraient être compensés par les entrées à l'intérieur du réseau ad hoc.

4.8.7 Fusion

Lors de la fusion, deux groupes indépendants se joignent pour ne plus former qu'un seul groupe. Un sponsor est élu dans chacun des groupes. L'un d'eux ou les deux peuvent envoyer une requête de fusion.

4.8.7.1 Opération chez les sponsors

- Chaque sponsor choisit une nouvelle contribution et calcule toutes les clés secrètes et cachées sur son chemin jusqu'à la racine.
- Il diffuse l'arbre des clés cachées obtenu à la suite de cette opération.
- Il détermine la hauteur de chacun des arbres
- Il construit un nouvel arbre dont chacun des deux arbres précédents constituent un sous-arbre. Le plus profond devient le sous arbre de gauche et l'autre, celui de droite.
- Dépendant du seuil d'activité, il complète l'équilibre du sous-arbre de droite.
- Il calcule la clé de groupe.

4.8.7.2 Opération au niveau des autres participants

- Ils déterminent la profondeur de chacun des arbres.
- Ils construisent un nouvel arbre dont chacun des deux arbres précédents constituent un sous-arbre. Le plus profond devient le sous-arbre de gauche et l'autre, celui de droite.

- Dépendant du seuil d'activité, ils complètent l'équilibre du sous arbre de droite.
- Il calcule la clé de groupe.

4.8.8 Le partitionnement

Lors du partitionnement, un sous-groupe de participants se sépare. L'opération se déroule comme suit :

- Le sponsor de ce sous-groupe envoie un paquet PARTITION contenant le sous arbre de clés des membres désirant se séparer.
- Le participant le plus à proche du sous-groupe sortant devient le sponsor du groupe restant.
- Le sponsor de chaque nouveau groupe change sa clé secrète et calcule les clés cachées de tous les nœuds parents sur son chemin et diffuse un message **update_keys** contenant un paquet BKEYS aux membres de son groupe

4.8.9 Le Rafraîchissement des clés

Le rafraîchissement survient lorsqu'il devient nécessaire de changer la clé de groupe pour préserver la confidentialité du passé, mais également la collusion. Il peut être effectué par chaque membre du groupe après la durée limite d'utilisation de la clé. Un paquet REFRESH est alors diffusé aux autres participants. Dans notre protocole, le rafraîchissement de la clé est demandé seulement lorsqu' aucun événement de groupe n'est pas survenu après une durée prédéfinie d'utilisation de la clé de groupe. Car en effet, si depuis la dernière génération de la clé de groupe il y a eu des événements de groupe (ajout, départ, etc.), la clé de groupe aura été changée et par conséquent le rafraîchissement de la clé est déjà réalisé. Par contre, si aucun événement de groupe n'aura eu lieu, la nécessité de rafraîchir la clé par l'envoie d'un paquet REFRESH s'impose après la durée prédéterminée pour le changement de la clé.

```
Void REFRESH_KEYS (sponsor_ID, BKEYS,Tree*)
{
        BKEYS * tab;
        Tab = &BKEYS,
                For (i=0;i<nb_key;i++)
                        Tree->noeud[i]->valeur =tab[i]->BKEY->valeur;
}
```

4.9 Conclusion

Dans ce chapitre, nous avons présenté la méthode GAKAP, une nouvelle méthode de distribution de clés de groupe multicast. Notre approche s'appuie sur la méthode TGDH et vise à réduire l'utilisation des ressources lors du calcul de la clé de groupe à la suite des événements de groupe.

Le protocole TGDH ayant été développé pour être utilisé lors des communications de groupe dans un réseau filaire, il n'offre pas une bonne performance dans les réseaux sans fil mobiles Ad Hoc. D'une part, dans ce type de réseau, la rupture des liens de communications est fréquente. Ces interruptions fréquentes de communication que l'on peut considérer comme des départs involontaires, ont une grande incidence sur l'équilibre de l'arbre des clés car elles entrainent de nombreuses séparations du groupe multicast pouvant aboutir à un déséquilibre important de l'arbre. D'autre part, le déséquilibre de l'arbre peut aussi survenir dans un groupe ou la prédominance des ajouts est marquée. Or, un arbre profondément déséquilibré nous donne une performance de l'ordre de O(n) pour le calcul de la clé de groupe en plus d'augmenter le nombre d'exponentiations totale effectuées par l'ensemble de membres du groupe et la consommation de la bande passante.

En introduisant l'usage de la constante d'activité, nous réduisons les risques de déséquilibre de l'arbre des clés lié aux départs multiples, aux séparations mais aussi à celui pouvant être crée en cas de prépondérance marquée des ajouts. Par la réduction de

ces déséquilibres, nous maintenons le nombre de rounds à l'ordre $\log_2 N$ ou N est les nombre de participants à la session multicast.
D'un autre côté, en utilisant les courbes elliptiques lors du calcul et des échanges de clés, nous pouvons réduire la consommation de la bande passante en réduisant la taille des messages transmis. Nous pouvons diminuer également le coût lié à la génération et au calcul des clés, donc de l'utilisation du processeur.

Le chapitre suivant présente la méthodologie utilisée pour réaliser les simulations nécessaires à l'évaluation des performances du protocole GAKAP.

Chapitre 5

Méthodologie de simulation

5.1 Introduction

Les simulations de faits, d'événements ou du comportement de diverses applications dans divers domaines de la vie font partie intégrante d'un ensemble d'outils permettant de comprendre et d'avoir une connaissance plus ou moins concrète des causes ou du résultat que l'on attend desdites applications.

Pour implanter ma méthode, j'ai utilisé la méthodologie de simulation. Les simulateurs de réseau offrent l'avantage de pouvoir créer des environnements complexes qui sont difficiles à mettre réellement sur pied pour des raisons de coût et de logistique, et d'en expérimenter virtuellement le comportement. Une **simulation** se définit donc comme l'exploration d'un modèle abstrait construit à partir d'observations du monde réel [40].

5.1.1 Définition

- NS-2 est un simulateur qui a été développé pour les simulations dans les réseaux filaires.
- Il a été étendu aux réseaux sans fil.
- Son code source a été développé en Tcl et C++ ce qui offre un compromis. intéressant quant à la performance et la facilité d'utilisation.
- L'implémentation et la simulation sous NS-2 consistent en 4 étapes :
    - L'implémentation du protocole en ajoutant du code C++ et Tcl au code source de NS-2.
    - La description du protocole en langage Tcl.
    - Les simulations proprement dites.
    - L'analyse des graphiques générés par le simulateur.

5.1.2 Désavantages :

- Absence de documentation à jour et limitée.
- Manque de consistance au niveau du code, certains protocoles fonctionnant dans les versions précédentes ne fonctionnent pas dans la nouvelle version.
- Manque d'outils généralisés d'analyse permettant de décrire les scénarios de simulation.
- Simulateur peu convivial comparativement à OPNET MODELER.

5.2 LES SIMULATIONS

5.2.1 Introduction

Pour la simulation, j'ai choisi un réseau ad hoc de 50 à 250 nœuds uniformément répartis sur une surface de 1000 m x 1000 m.

5.2.2 Modèle de mobilité

- La mobilité est une mesure importante dans le développement des protocoles destinés aux réseaux ad hoc.
- Elle affecte considérablement la connectivité des nœuds dans les réseaux sans fil mobiles.
- J'ai choisi le Random Waypoint comme modèle de mobilité à cause de sa popularité mais aussi à cause des mouvements aléatoires et indépendants entre les nœuds.

5.2.3 Métriques et paramètres

Pour déterminer la performance du protocole, nous avons définit 3 métriques :

- La première permet de mesurer le nombre d'échanges effectués lors du calcul de la clé.
- La deuxième permet de mesurer le temps de calcul de la clé.

- La troisième, permet de mesurer l'impact de la mobilité et de la fréquence des événements de groupe sur le calcul de la clé de groupe et la structure de l'arbre des clés.

5.2.3.1 Métriques

- Nombre total de messages diffusés.
- Nombre de rounds effectués pour calculer la clé de groupe.
- La valeur seuil de la constante d'activité.

5.2.3.2 Paramètres

Paramètres	Valeurs
Taille du terrain	1000 m x 1000m
Nombre de nœuds	50 à 250
Déploiement de nœuds	Uniforme
Nombre de nœuds émetteurs	1
Nombre de nœuds récepteurs	49
Protocole MAC	802.11
Taux de transfert	2Mbps
Model de propagation du signal radio	FreeSpace
Type d'antenne	Omnidirectionnelle
Modèle de mobilité	Random Waypoint
Vitesse minimale	2 m/s pause =10s

Nous avons une vitesse minimale non nulle pour éviter la décroissance de la vitesse des nœuds durant la simulation.

5.2.4 Inconvénients des simulations

- Une des limitations des simulateurs est due au fait que les applications sont restreintes à des modèles existants.
- Il y a risque de simplification des applications complexes car les simulations ne sont pas en adéquation parfaite avec la réalité, elles ne sauraient donc pas rendre compte des multiples complexités de l'application réelle d'un protocole.

5.2.5 Les étapes d'une simulation réseau

Les simulations réseau se font en cinq principales étapes : la définition du problème, la modélisation, la simulation proprement dite, l'analyse des résultats et la prise de décision.

5.2.6 Environnement de développement

Notre protocole a été développé en C/C++ sous linux. Cette plate forme a été choisie pour 2 raisons :

- Le simulateur NS-2 a été développé sous linux et s'y intègre donc facilement.
- Le protocole TGDH dont nous nous inspirons a été implémenté en C sous linux.

5.2.7 Précision des simulations

Pour qu'une simulation soit valide, il faut que l'application ou le protocole que l'on simule colle à la réalité, c'est-à-dire qu'il faut qu'il se fonde sur un modèle déjà existant. Il faut en outre que les paramètres de la simulation soient réalistes car le choix de mauvais paramètres de départ pourraient considérablement biaiser les résultats.

5.2.8 La constante d'activité

La constant d'activité a été déterminer comme étant le rapport de deux paramètres obéissant à la lois de Poisson. Pour la calculer nous avons :

- Déterminer par expérience une valeur seuil s_0 pour laquelle le système offre le meilleur équilibre.
- Déterminer le nombre d'ajouts selon une loi de poisson de paramètre λ.
- Déterminer le nombre de départs selon une loi de poisson de paramètre μ.
- Chercher la valeur $\Omega = \mu/\lambda$ tel que $\Omega \approx s_0$.
- Fixer une valeur k pour laquelle si $\Omega > k$ alors le système de distribution de clé est considéré comme déséquilibré.

5.2.9 Implémentation de la méthode

L'implémentation de la technique GAKAP s'est faite en trois étapes :

- L'apprentissage du simulateur
- L'implémentation du protocole en langage C/C++ et l'interfaçage en langage TCL
- L'intégration du code développé dans le simulateur NS-2

Le développement du protocole s'est donc fait en deux étapes, la première consistant à l'implémenter dans l'environnement de développement KDevelop et la deuxième, à l'implémenter dans le simulateur NS-2.

5.2.9.1 Implémentation en C/C++ du protocole GAKAP

Le programme en C/C++ du protocole utilise plusieurs structures et fonctions. Une structure permet de représenter les participants lors de la session multicast, l'arbre des clés, les nœuds de l'arbre et les clés associées à chaque nœud. Plusieurs fonctions ont été définies et permettent de calculer la taille et la profondeur de l'arbre de clés, de générer aléatoirement le type d'événement de groupe survenu. Lorsque survient un événement de groupe, le comportement des différents paramètres que l'on veut mesurer est passé à des variables définies dans le code.

Le programme compare les protocoles TGDH et GAKAP du point de vue de leur capacité à s'adapter à un groupe très dynamique. La constante d'activité est obtenue

dans le programme comme valeur en tant que ratio entre le nombre de départs et le nombre d'ajouts.

Le code que j'ai écrit se fonde sur les spécifications des protocoles TGDH et GAKAP. Son intérêt est de faire ressortir l'impact de la fréquence des ajouts et des départs sur l'équilibre de l'arbre des clés et de son incidence sur le nombre de rounds qui sont effectués pour parvenir au calcul de la clé de groupe. Pour y arriver, j'ai défini :

- la fonction **tree_deep** qui donne la profondeur de l'arbre en fonction du nombre de participants et cette profondeur varie avec les événements qui surviennent dans le groupe.
- La fonction **nb_free_nodes** qui donne le nombre de nœuds vides à combler.
- La fonction **is_node_free** qui vérifié si le nœud est libre d'occupation.
- D'autres fonctions comme **DelPart**, **DelParts**, **AddPart**, **AddParts**, **ReconnectPart** ou **DisconnectPart** ont pour but d'effectuer les événements courants de groupe.

5.2.9.2 Implémentation sous NS

L'implémentation dans le simulateur NS-2 s'est effectuée en plusieurs étapes et a nécessité la modification de certaines parties du code contenues dans le simulateur.

5.2.9.2.1 Définition du protocole

Tout d'abord, il a fallu intégrer le protocole GAKAP au sein des paquets existant dans le simulateur. On a incorporer les structures et objets définissant le protocole GAKAP parmi les objets et structures du simulateur de la façon suivante :

- On défini la structure **struct hdr_gakap** correspondant à l'entête du protocole GAKAP
- Ensuite on a défini une classe d'objet **GakapAgent** correspondant à l'agent du protocole chargé de transmettre et de recevoir les paquets du protocole. Cet

agent est celui qui sera instantié par les différents nœuds mobiles représentant les participants à la session multicast.

- En troisième lieu, on a construit la classe **GakapClass** dont les objets permettent de servir d'interface aux objets qui seront définis en Tcl et qui appelleront l'exécution du code définit en C/C++. Cette classe se définit comme suit :

```
static class GakapClass : public TclClass {
public:
  GakapClass() : TclClass("Agent/Gakap") {}
  TclObject* create(int, const char*const*) {
    return (new GakapAgent());
  }
    } class_gakap;
```

- Par la suite on a modifié certaines structures du code interne du simulateur NS pour lui spécifier l'existence d'un nouveau protocole.
- Enfin on a écrit un programme en Tcl définissant un réseau mobile ad hoc constitué de nœuds exécutant le protocole GAKAP.
- Pour contrôler le flux des paquets ADD ou ADD_M et LEAVE ou LEAVE_M, on a utilisé les agents de la classe **CBR** (Contant Bit Rate agents) de NS-2 afin de regarder le comportement de l'arbre des clés.
- On a utilisé les objets de la classe **mobilenode** de NS-2 pour représenter chaque participant.

La procédure ci-dessus a été également effectuée pour le protocole TGDH. Les courbes obtenues proviennent du programme xgraph qui vient avec le simulateur NS.

5.3 Conclusion

Dans ce chapitre, nous avons présenté les avantages qu'apportent les simulateurs réseau du point de vue des économies financières, mais également de la modélisation presque réelle des applications dans un environnement virtuel.

Le simulateur NS a été choisi pour plusieurs raisons. D'abord, NS est gratuit et permet de réaliser des simulations complexes. C'est l'un des simulateurs les plus utilisés au

niveau des simulations réseau. Mais le véritable problème avec NS est son manque de convivialité, il n'offre pas d'environnement graphique comme OPNET et il faut aussi avoir une assez bonne connaissance des systèmes unix pour l'utiliser, car toutes les commandes se font au niveau de la console de commandes. Le code en C/C++ est assez compréhensible et facilite la réutilisation (l'héritage) des classes d'objets. Nous avons utilisé les classes **Mobilenode**, **Event**, **Agent** pour définir nos objets.

Le chapitre suivant présente les différents résultats de simulation. Une comparaison entre les méthodes TGDH et GAKAP y est détaillée.

CHAPITRE 6

Résultats

6.1 Introduction

Au chapitre précédent j'ai présenté ce que sont les simulations réseau, leurs avantages et leurs inconvénients, j'ai également montré la façon dont je me suis pris pour réaliser les simulations des protocoles GAKAP et TGDH. J'ai utilisé l'environnement KDevelop du système Linux pour développer le code en C/C++. Ce code a été par la suite adapté et intégré au simulateur NS-2.29.

6.2 Résultats des simulations

6.2.1 Résultats du code C/C++

Pour étudier le comportement des protocoles TGDH et GAKAP, j'ai développé un programme en C/C++. Les événements sont générés aléatoirement, les ajouts et les départs sont générés selon des intervalles de temps différents de façon à mesurer l'effet du grand nombre d'ajouts ou de départs sur l'équilibre de l'arbre des clés, le nombre de messages transmis par le sponsor, le nombre de rounds effectués pour calculer la clé de groupe. Pour mesurer le comportement des différents paramètres énoncés ci-dessus, le nombre de rounds, le nombre de paquets transmis, j'ai fait varier simultanément le nombre de participants et le nombre d'événements initiaux de la façon suivante (tableau VI) :

Nombre de participants	Nombre d'événements
25	10
50	20
75	30
100	40

125	50
150	60
175	70
200	80
225	90
250	100

Tableau VI: Nombre d'événements en fonction du nombre de participants.

Ensuite, j'ai maintenu de façon constante le nombre de participants à 250 (le maximum de participants possible) et j'ai fait varier le nombre d'événements de 10 à 100. Une fois le programme en cours, il génère lui-même des événements dont le nombre est saisi dans des variables. Chaque fois, le temps de simulation de chacun des cas était de deux ou de trois minutes, et les résultats ci-dessous ont été obtenus.

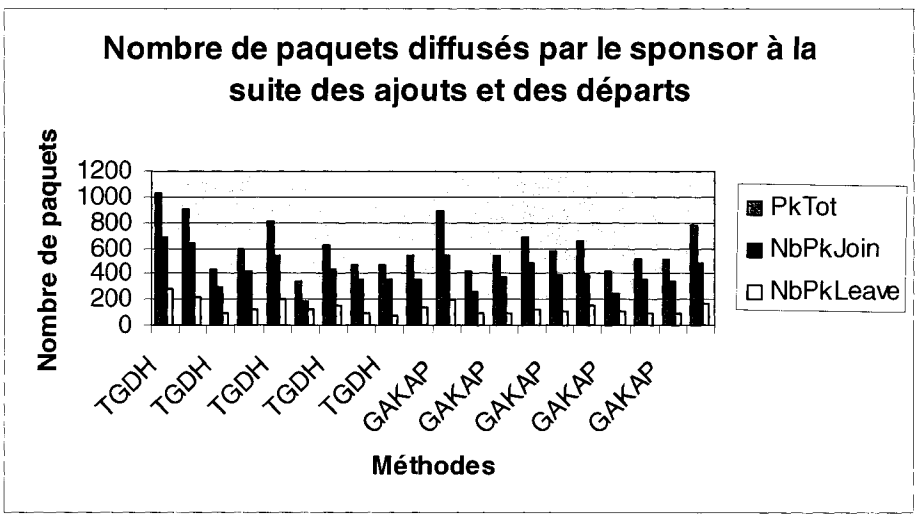

Figure 32 : Nombre de paquets diffusés par les sponsors suite aux ajouts et départs des participants.

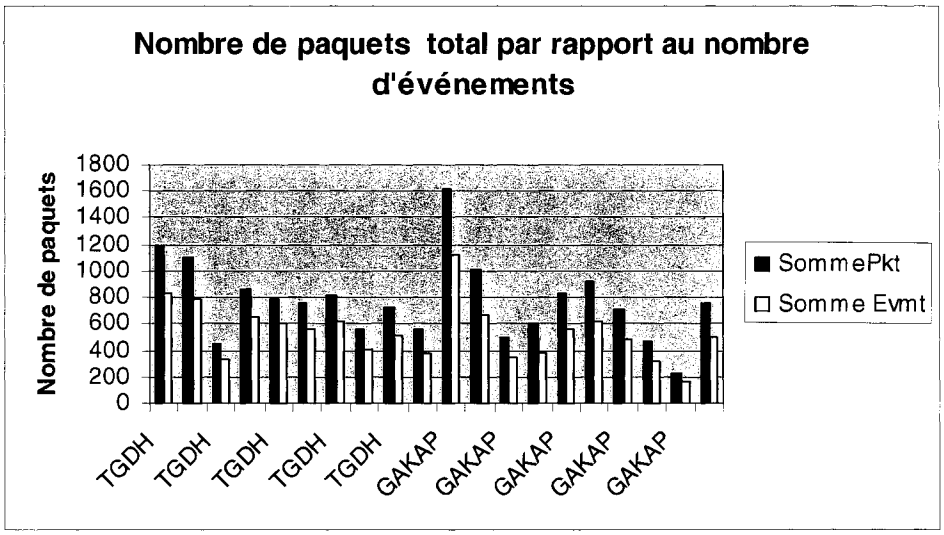

Figure 33 : Nombre total de paquets lors des ajouts et départs par rapport à la somme des événements.

Dans les figures 32 et 33, les dix premiers groupes de bâtonnets correspondent aux valeurs de TGDH et les dix derniers à GAKAP. Ce sont, le nombre de paquets total diffusés (PkTot), le nombre de paquets d'ajout (NbPkJoin) et de départ (NbPkLeave), la somme des paquets (somme Pkt) et la somme des événements (somme Evmt). On observe que pour des valeurs équivalentes de nombres de participants et selon le nombre d'événements d'ajouts ou de départs survenus dans le groupe :

1. La figure 32 montre que dans un environnement très dynamique, le sponsor dans le protocole GAKAP diffuse moins de paquets lors des événements d'ajout et de départ pris ensemble.
2. la figure 33 permet de constater que le ratio du nombre de paquets par rapport au nombre total d'événements est aussi inférieur dans le protocole GAKAP.

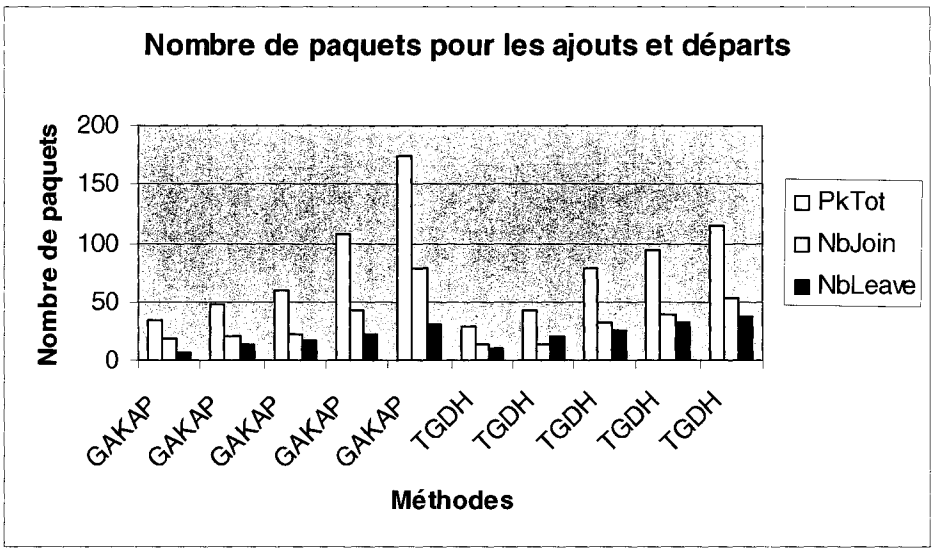

Figure 34 : Nombre de paquets total envoyés lors des ajouts et départs

La figure 34 montre que le nombre de paquets total (PkTot) envoyés par le ou les sponsors lors des événements associés d'ajout et départ est plus important pour le protocole GAKAP. Ceci s'explique par le fait que lors des ajouts simples, le sponsor envoie deux types de paquets, un paquet unicast au nouveau membre et un paquet multicast au groupe.

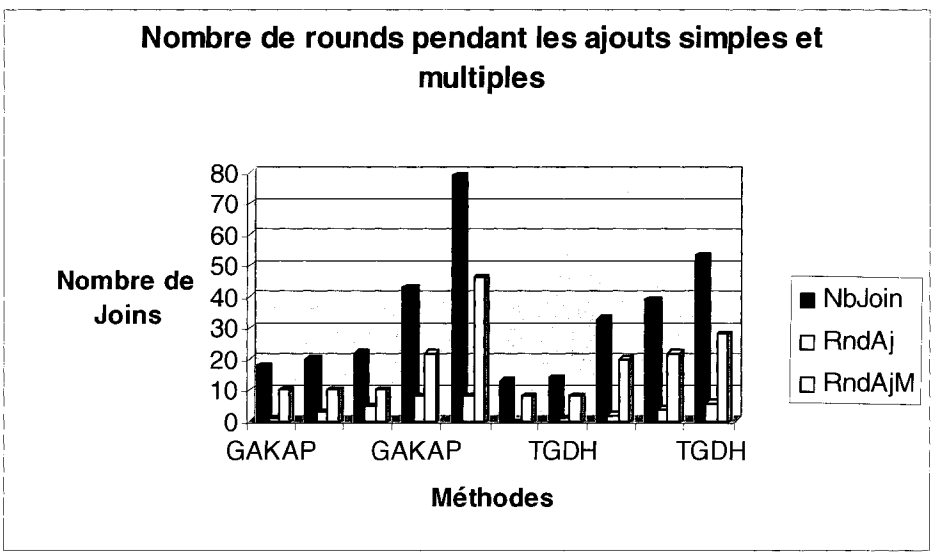

Figure 35 : Nombre de rounds à la suite des ajouts simples et multiples

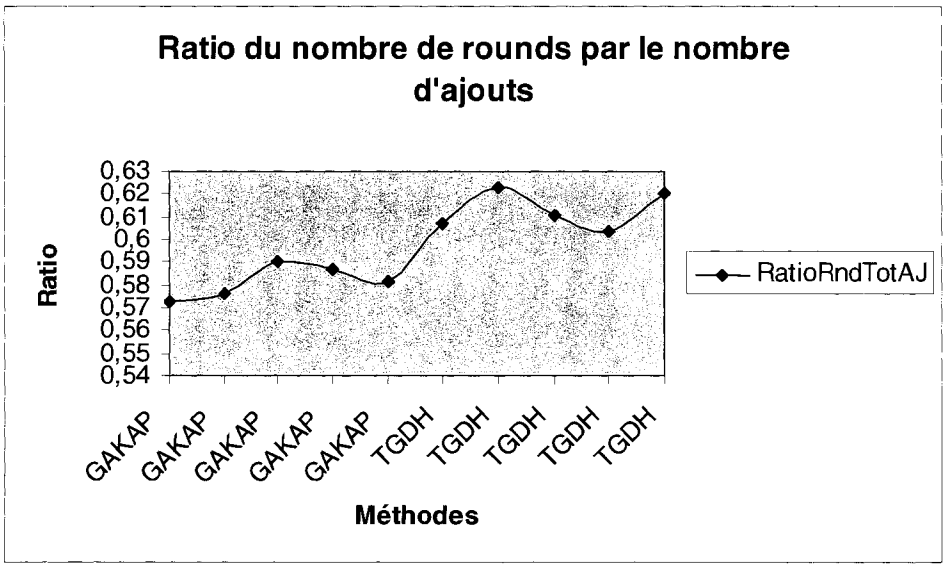

Figure 36 : Rapport du nombre de rounds par rapport au nombre total d'ajouts

Les figures 35 et 36 présentent respectivement le nombre de rounds qui sont effectués pour le calcul des clés lors des ajouts simples et multiples mis ensemble et le ratio obtenu en calculant le rapport de la somme du nombre de rounds effectués pour les ajouts simples (RndAj) et multiples (RndAjM) par le nombre d'événements. Les On remarque que :

- le nombre de rounds croît avec le nombre d'ajouts.
- Le nombre de rounds croît avec le nombre d'événements par unité de temps.
- le ratio du nombre de rounds par rapport au nombre d'ajouts est inférieur dans le protocole GAKAP.

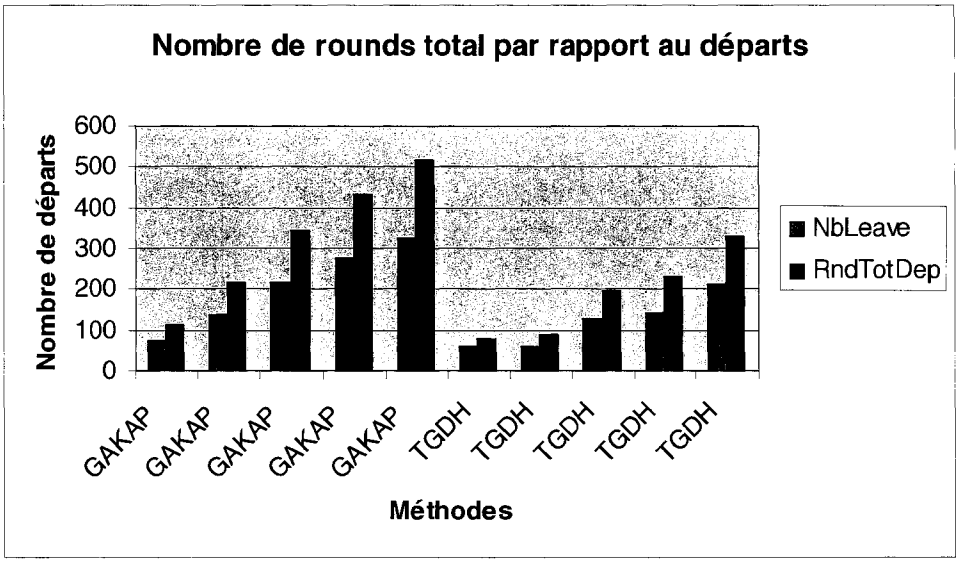

Figure 37 : Nombre de rounds total à la suite des départs simples et multiples

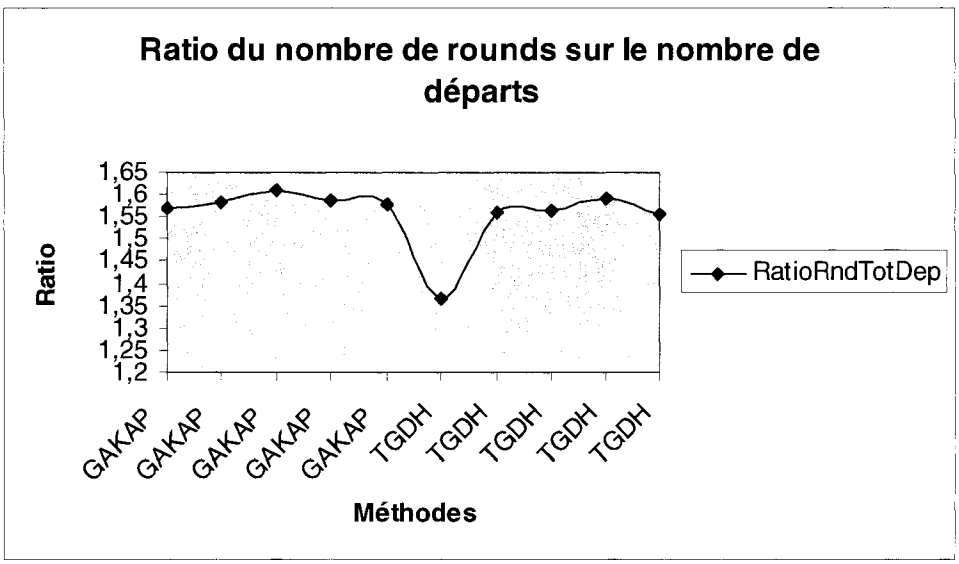

Figure 38 : Rapport du nombre de rounds par rapport au nombre total de départs.

Tout comme au niveau de l'ajout, les figures 37 et 38 présentent respectivement le nombre de rounds qui sont effectués pour le calcul des clés lors des départs simples et multiples, et le ratio obtenu en calculant le rapport de la somme du nombre de rounds effectués pour les départs simples et multiples par le nombre d'événements. On remarque que :

- le nombre de rounds croît avec le nombre de départs.
- le ratio du nombre de rounds par rapport au nombre de départs est légèrement supérieur dans le protocole GAKAP.

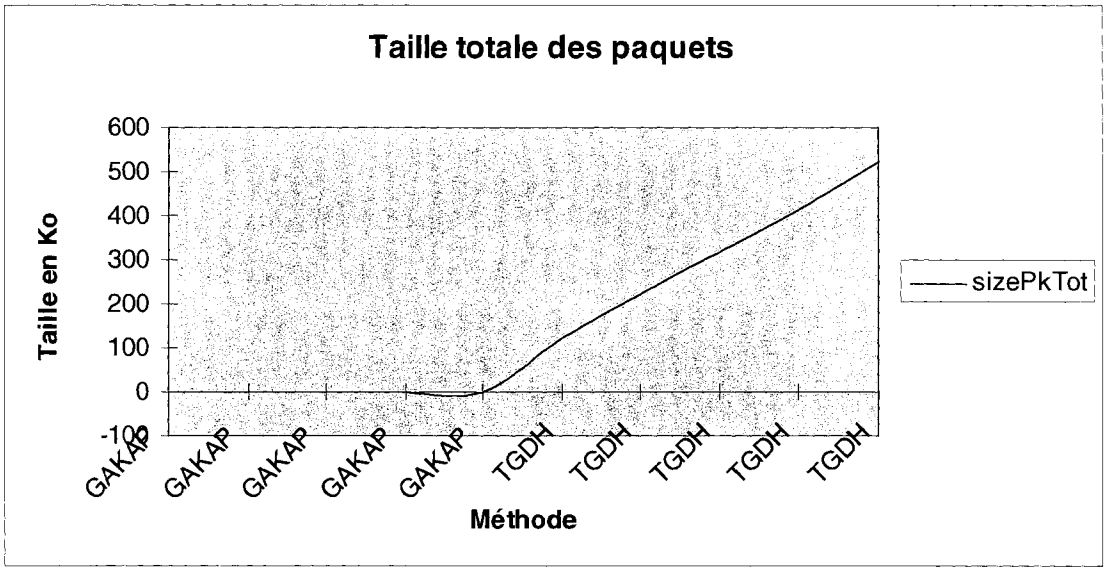

Figure 39 : Taille totale des paquets envoyés lors de tous les événements d'ajout et départ.

La figure 39 montre le nombre de paquets envoyés par les sponsors lors de tous les événements survenus pendant la simulation. On constate que :

- Malgré la quantité plus faible du nombre de paquet émis par le protocole TGDH, la taille des paquets envoyés par le protocole GAKAP constitue une trace alors que le nombre de paquets produits avec TGDH croît de façon importante.
- Cela s'explique d'abord parce que la taille des clés de chiffrement produites dans GAKAP est inférieure mais aussi parce que lors des événements de groupe, le sponsor envoie seulement les clés ayant été modifiées alors que dans TGDH, tout l'arbre des clés est acheminé.

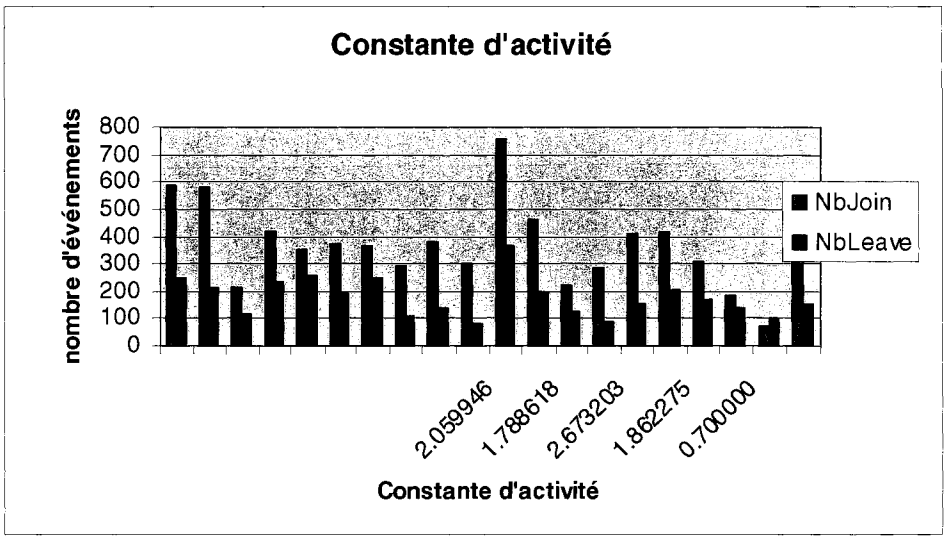

Figure 40 : La constante d'activité en fonction du nombre d'ajouts et de départs.

Pour calculer la constante d'activité, j'ai fait varié le nombre de membre de 25 à 250, le nombre d'événement a été généré par le système. La figure 40 montre que dans un environnement très dynamique, il faut une constante d'activité d'environ 1,8 à 1,9 ajout pour un départ pour maintenir un équilibre équivalent entre TGDH et GAKAP, c'est-à-dire un nombre d'ajouts et de départs qui malgré les changements de structure de l'arbre des clés, maintient un équilibre au niveau des rounds à effectuer pour obtenir la clé de groupe. Il faut noter que la constante d'activité est uniquement exprimée ici pour le protocole GAKAP et correspond aux dix valeurs de droite de la figure 40, les valeurs de gauche étant tout simplement le nombre d'ajout et de départ pour le protocole TGDH. Seules les cinq valeurs les plus significatives apparaissent sur le schéma.

6.3 Conclusion

Dans ce chapitre, nous avons procédé à l'évaluation des protocoles TGDH et GAKAP dans un environnement très dynamique. Pour ce faire, nous avons écrit un programme en C/C++ qui nous a permis d'évaluer les paramètres suivants : le nombre de paquets diffusés par les sponsors, le nombre total de paquets par rapport à la somme des événements, le nombre de paquets total envoyés lors des événements de groupe, la

taille totale des paquets envoyés lors de tous les événements, le nombre de rounds lors des événements de groupe et la constante d'activité. Nous avons constaté que le protocole GAKAP permet une meilleure consommation de la bande passante grâce à la taille réduite des messages, un meilleur coût lors du calcul des clés, offre un ratio du nombre de rounds lors des départs égal à celui de TGDH et permet de déterminer le rapport du nombre d'ajouts et de départ pour lequel l'équilibre de l'arbre des clés est préservé et le coût relié au calcul de la clé de groupe maintenu à l'ordre O(logN).

Dans le chapitre qui suit, nous allons présenter la conclusion et les perspectives à la recherche que nous avons amorcée dans le cadre de ce travail.

CHAPITRE 7

TRAVAIL EN COURS ET PERPECTIVES

7.1 Travail en cours

Dans les chapitres 5 et 6, j'ai présenté le cadre à travers lequel je devais réaliser les simulations qui viendront valider mes hypothèses de recherche. J'ai développé un programme C/C++ qui m'a permis de simuler le comportement des protocoles GAKAP et TGDH dans un environnement ou les événements de groupe sont fréquents et les résultats de cette simulation apparaissent au chapitre 6. L'insertion de mon code au sein du simulateur NS s'est avérée quant à elle assez complexe à cause :

- De la complexité à trouver de l'information qui m'aurait guider à implémenter un protocole distribution de clé dans un environnement mobile.
- De la difficulté à modifier toutes les structures nécessaires pour définir toutes les spécifications du protocole GAKAP relatives à la mobilité des nœuds.
- Du manque de temps pour l'apprentissage du langage TCL et de l'interfaçage qu'il permet avec l'environnement de simulation.

Mon travail s'oriente donc présentement vers l'intégration du code dans le simulateur NS. Les résultats obtenus à la suite de ces simulations feront l'objet d'une présentation pour de prochaines publications.

7.2 Perspectives

Les réseaux sans fil constituent un domaine de recherche en plein développement. Ce sont des réseaux ou la sécurité est encore précaire comparativement aux réseaux filaires. Avec le développement des applications de groupe comme la vidéoconférence, le télétravail, les technologies sans fil et le multicast seront de plus en plus sollicités. La sécurité des applications de groupe, dont la gestion des clés de cryptage demandera à être optimisée.

Dans les réseaux sans fil, l'un des principaux problèmes est le manque de ressources, principalement la bande passante, la faiblesse des processeurs et la limitation de la consommation énergétique des périphériques. Pour pallier le manque de ressources, une des pistes de solution serait des méthodes de cryptage basées sur l'utilisation des courbes elliptiques lesquelles permettent de réduire la taille des clés. Dans les réseaux à grande mobilité, il serait judicieux de déterminer des paramètres de vitesse et de la dynamique de groupe multicast qui permettrait de gérer l'équilibre de l'arbre des clés, de minimiser le nombre de clés à transmettre, compte tenu du fait que la mobilité provoque d'importantes ruptures de liens et si ces liens ne sont pas récupérées, cela augmentera les cas de départs « involontaires ».

De plus, tous les nœuds du réseau ad hoc participent à la tâche de routage, ils génèrent donc un trafic supplémentaire dans le réseau. La réduction de la taille des clés et du nombre de transmissions par le rééquilibrage de la structure de l'arbre des clés est une piste dans l'utilisation des méthodes de distribution des clés de cryptage en arbre pour les réseaux mobiles ad hoc sans fil.

Ma méthode permet de déterminer le niveau de mobilité et le ratio d'événements de groupe pour lequel on maintient un nombre de distribution de clés qui n'excède pas $O(\log_2 N)$.

L'un des principaux handicape de ma méthode est la difficulté à déterminer des courbes elliptiques efficaces. De plus, il n'est pas encore prouvé que les courbes elliptiques offrent véritablement toute la sécurité qu'on leur attribue présentement. Il serait donc important que les recherches futures se penchent sur les méthodes de distribution de clés basées sur l'utilisation des courbes elliptiques.

CONCLUSION

A leurs débuts, les technologies du Multicast ont été développées pour l'utilisation de ressources, en particulier celle de la bande passante lors des communications impliquant de multiples hôtes. En effet, avec ces technologies, un hôte peut émettre un unique paquet vers plusieurs autres hôtes et réduire ainsi l'utilisation de sa bande passante. Ceci devient doublement plus important lorsqu'il s'agit d'échange de clés de chiffrement dont on sait que la taille est généralement grande.

Dans ce travail, j'ai proposé une nouvelle méthode de distribution des clés de chiffrement lors des communications multicast dans un réseau MANET très dynamique tant au niveau des événements de groupe que de celui de la mobilité. Ce protocole a été appelé GAKAP et visait principalement à réduire le nombre de paquets diffusés, le nombre de rounds à effectuer pour le calcul de la clé de groupe lors des ajouts et des départs fréquents des participants à cause des ruptures possibles de communications dues à la mobilité.

Ce travail de recherche est une contribution aux différents problèmes qui se posent présentement dans le domaine du Multicast et qui entravent son adoption et son application effective dans l'industrie. Cet apport consistait à orienter le questionnement sur la viabilité et l'efficacité du protocole TGDH et d'autres protocoles distribution de clés multicast dans les réseaux sans fil mobile.

Compte tenu des résultats obtenus lors des simulations, la méthode GAKAP donnerait de meilleurs résultats que la méthode TGDH dans un réseau mobile ad hoc très dynamique. En effet, la quantité d'informations réellement échangées est plus bas avec la méthode GAKAP qu'avec la méthode TGDH. Malgré le fait le protocole GAKAP utilise des envois unicast lors de l'ajout des nouveaux participants, on s'aperçoit que

globalement le ratio de paquets diffusés lors des événements de groupe est plus important avec la méthode TGDH.
Mon objectif était également de réduire la taille des clés de chiffrement et le temps de calcul, l'utilisation du chiffrement par courbe elliptique a permis d'atteindre ce double objectif.
Non seulement la méthode utilise moins de paquets lors des ajouts et des départs des participants, mais elle utilise en moyenne moins de rounds lors du calcul de la clé de groupe.

L'une des difficultés majeures dans la réalisation de ce projet a été l'apprentissage du simulateur NS et de la façon dont il fallait se prendre pour intégrer notre protocole au simulateur. D'autres difficultés ont été d'apprendre et de comprendre le langage TCL utilisé comme interface du code en C pour les simulations sous NS, d'écrire le programme C/C++ du protocole qui tienne en compte les aspects liés à la mobilité des nœuds en suivant les modèles de mobilité déjà existant. Une recherche ardue m'a permis de constater que les structures de données définies dans le simulateur NS permettaient l'ajout de nouvelles constantes et paramètres définissant le nouveau protocole.

BIBLIOGRAPHIE

[1] D. Wallner, E. Harder et R. Agee, **Key management for multicast: issues and architecture**, Internet Engineering Task Force, no. 2627, Jun. 1999.

[2] A. Ballardie, **Scalable Multicast Key Distribution**, Internet Engineering Task Force, no. 1949, May 1996.

[3] David A. McGrew et Alan T. Sherman, **Key establishment in large dynamic groups using one-way function trees**, mai 1998

[4] H. Harney et C. Muckenhirn, **Group key management protocol (GKMP) architecture**. RFC 2094, July 1997.

[5] H. Harney et C. Muckenhirn, **Group key management protocol (GKMP) specification**. RFC 2093, July 1997.

[6] Danilo Bruschi et Emilia Rosti. **Secure Multicast in Wireless Networks of Mobile Hosts: protocols and issues**, mobile ad hoc network and application, vol.7, no6, dec. 2002

[7] Paul Judge et Mostafa Ammar. **Security Issues and Solutions in Multicast Content Distribution: A survey**. IEEE Network, vol.17,no 1, january-february 2003

[8] T. Harjono et Gene Tsudik. **Ip Multicast Security: Issues and Directions**, annales de telecom,pp.324-334, july-august 2000

[9] Yongdae Kim, Adrian Perrig, et Gene Tsudik. **Simple and fault-tolerant key agreement for dynamic collaborative groups**. *In* Proceedings of the 7th ACM conference on Computer and communications security, *pages 235-244, 2000*

[10] Eric Anton et Otto Duarte. **Group key establishment in wireless ad hoc networks**. In Workshop on Quality of service and mobility, 2002.

[11] Maria Striki, John S.Baras. **Key Distribution Protocols for Multicast Group Communication in MANETs**, ISR Technical Reports, no TR 2003-17, 2003

[12] Xiang-Yang Li. Yu Wang. Ophir Frieder **Efficient hybrid key agreement protocol for wireless ad hoc networks** Computer Communications and Networks, 2002. Proceedings of the Eleventh International Conference on, 14-16 Oct. 2002 Pages:404 - 409

[13] John.S.Baras, M.Striki. **Two Level Hybrid Key Scheme for Efficient Key Distribution in Mobile Ad-Hoc Networks** Récupéré le 18 juillet 2003 de http://www.asc2002.com/summaries/o/OP-22.pdf

[14] M.Striki, John.S.Baras. **Scalable and Efficient Key Agreement Protocols for Secure Multicast Communication in MANETs**, ISR Technical Reports,no TR 2002-62, 2002 récupéré le 18 juillet 2003 de http://techreports.isr.umd.edu/reports/2002/TR_2002-62.pdf

[15] K.Becker, U.Wille. **Communication complexity of group key distribution**. In Proceedings of the 5th ACM conference on Computer and communications security, novembre1998

[16] M. Steiner, G. Tsudik, M. Waidner. **Diffie-Hellman Key Distribution Exteded to Group Communication**. In Proceedings of the 3rd ACM conference on Computer and communications security, janvier 1996

[17] A.Selcuk, C. McCubbin and D. Sidhu, "**Probabilistic Optimization of LKH-based Multicast Key Distribution Schemes,**" Internet Engineering Task Force, Jan. 2000.

[18] D. Balenson, D. McGrew and A. Sherman, "**Key Management for Large Dynamic Groups: One-Way Function Trees and Amortized Initialization,**" Internet Engineering Task Force, Sep. 2000.

[19] T. Hardjono, B. Cain and N. Doraswamy, "**A Framework for Group Key Management for Multicast Security,**" Internet Engineering Task Force, Aug. 2000.

[20] Chung Kei Wong, Mohamed Gouda, and Simon S. Lam, "**Secure group communications using key graphs,**" in Proceedings of ACM SIGCOMM '98, September 1998.

[21] R. Canetti and B. Pinkas, "**A taxonomy of multicast security issues,**" Internet Engineering Task Force, Aug. 2000.

[22] Ran Canetti, Juan Garay, Gene Itkis, Daniele Micciancio, Moni Naor and Benny Pinkas, "**Multicast Security: A Taxonomy and Efficient Constructions,**" in Proceedings of the Conference on Computer Communications (IEEE Infocom), (New York), Mar. 1999.

[23] C. Kenneth MILLER, **Multicast Networking And Applications**, Addison Wesley, 1998

[24] K. Al Agha et G. Pujolle, G. Vivier, **Réseaux de mobiles & réseaux sans fil**, Eyrolles, 2001

[25] L. Buttyan, J-P. Hubaux, **Report on Working Session on Security inWireless Ad Hoc Networks**, Mobile Computing and Communication Review, Volume 7, number 1

[26] Imrich Chlamtac, M.Conti, Jennifer J-N. Liu, **Mobile ad hoc networking: imperatives and challenges**, Elsevier , 2003

[27] Sandro Rafaeli, David Hutchison, **A Survey of key management for secure group communication**, in ACM Computing surveys, volume 35, No 3, September 2003,pp.309-329.

[28] Varshney Upkar, **Multicast Over Wireless Networks**, in the communication of the ACM, volume 45, No 12, December 2002.

[29] Maarit Hietalahti, **Efficient Key Agreement for Ad-hoc Networks,** Master's Thesis, Helsinki University of technology, may 23, 2001

[30] S. Ruffino, P. Stupar, T. Clausen, **Autoconfiguration in MANET: connectivity scenarios and technical issues,** Internet Draft, October 2004

[31] M. Aydos, T. Yanik, and C. K. Koc. **High-speed implementation of an ECC-based wireless authentication protocol on an ARM microprocessor**. *IEEE Proceedings - Communications,* 148(5):273-279, October 2001.

[32] Vipul Gupta, Douglas Stebila, Stephen Fung, Sheueling Chang, Nils Gura, and Hans Eberle. **Speeding up Secure Web Transactions using Elliptic Curve Cryptography**. 11th Ann. Symp. on Network and Distributed System Security — NDSS 2004, Internet Society, February, 2004.

[33] Manoj Pandey and Daniel Zappala, **The Effects of Mobility on Multicast Routing in Ad Hoc Networks**. March, 2004 (Technical Report, UO-CIS-TR-2004-2).

[34] Y. Amir, Y. Kim, C. Nita-Rotaru, G. Tsudik, **On the performance of Group Key Agreement Protocols**, ACM Transactions on Information and System Security (TISSEC), Volume 7 , Août 2004, pp. 457 - 488

[35] F. Bai, N. Sadogopan, A. Helmy. **IMPORTANT: A framework to systematically analyze the Impact of Mobility on Performance of Routing protocols for Adhoc Networks.**In IEEE INFOCOM, 2003

[36] W. Diffie et M. Hellman: **New directions in cryptography.** IEEE Transactions on Information Theory, 22(6), 1976, P. 644-654.

[37] Y. Kim, A. Perrig,G. Tsudik, **Tree-Based Group Key Agreement.** Récupéré en Décembre 2003 de http://www-users.cs.umn.edu/~kyd/paper/kpt02.pdf

[38] T. Wollinger, J. Pelzl, V. Wittelsberger, C. Paar, G. Saldamli, and C. K. Koc. **Elliptic and hyperelliptic curves on embedded μP.** *ACM Transactions on Embedded Computing Systems,* 3(3):509-533, August 2004.

[39] Don B. Johnson, Alfred J. Menezes, **Elliptic Curve DSA (ECDSA): An Enhanced DSA.** Récupéré le 20 mai 2005 de http://islab.oregonstate.edu/koc/ece575/papers/ecdsa.pdf .

[40] Sally Fyold, Vern Paxon, **« Difficulties in simulating the internet »**, IEEE/ACM, Transaction on networking, vol 9, no 4, pages. 392-403, Août 2001.

[41] Marc Joye, **Introduction élémentaire à la théorie des courbes elliptiques**, Technical Report CG-1995, juin 1995.

[42] Sandro Rafaeli, **A Decentralised Architecture for Group Key Management**, Récupéré le 12 septembre 2003 de

[43] Roberto Di Pietro, Luigi V. Mancini, Sushil Jajodia, **Efficient and Secure Keys Management for Wireless Mobile Communications**, POMC'02, 30 Octobre, 2002.

[44] Sandro Rafaeli, Laurent Mathy et David Hutchison. **"EHBT: an efficient protocol for group key management"**, International Workshop on Networked Group Communications 2001, pages 159-171, 2001.

[45] M. Steiner, G. Tsudik et M. Waidner. **"Cliques: A new approach to group key agreement"**, IEEE Transactions on Parallel and Distributed Systems, pages 380-387, 1998.

[46] S. Mittra, **"Iolus: A framework for scalable secure Multicasting"**, Proceedings of ACM SIGCOMM '97, pages 277–288, Cannes, France, septembre 1997.

ANNEXE 1: CODE DE LA SIMULATION

Le Fichier contenant le code en C/C++ de la simulation est disponible au niveau du laboratoire LAMIA.

ANNEXE 2: PUBLICATIONS

A. Boumso, B. Amar Bensaber, I. Biskri, « Multicast Key Agreement Protocol for Mobile Ad Hoc Networks», 4ème Conférence sur la Sécurité et Architectures Réseaux (SAR'05), Batz sur Mer - Loire Atlantique, France, 6-10 juin 2005.

A. Boumso, B. Amar Bensaber, I. Biskri, « GAKAP, Multicast Key Agreement Protocol for Ad Hoc Networks Based On Group Activity Probability», 4th International IEEE Workshop on Wireless Local Networks, Tampa, Florida, November 04. *Cet article, présenté dans une conférence internationale IEEE, traite le problème de la sécurité des communications Multicast sur les réseaux sans fil Ad Hoc mobile. L'accent est mis sur la gestion de l'arbre des clés en fonction de l'activité du groupe.*

M. St Amand, B. Amar Bensaber, **A. Boumso** « ROKeN-SKI: a pragmatic solution for Key Distribution in Secure Multicast Communications », *Proceedings of the* Mediterranean Workshop on Ad-Hoc Networks, Mahdia, Tunisia, 25-27 June, 2003.

ANNEXE 3 : COMMUNICATIONS

A. BOUMSO, B. AMAR BENSABER, «Méthode de distribution de clés de cryptage pour les communications Multicast dans les réseaux sans fil mobile ad hoc». Congrès Gestion, innovation et technologie de l'information, 72ème congrès de l'ACFAS, Montréal 2004.